

# **RANCANG BANGUN 3D ORIENTED MOUSE CONTROLLER MENGUNAKAN FINITE STATE MACHINE BERBASIS MIKROKONTROLLER ARDUINO**

## **SKRIPSI**

Untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Teknik

Disusun oleh :  
Firdy Yantama  
NIM: 145150300111049



**PROGRAM STUDI TEKNIK KOMPUTER  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS BRAWIJAYA  
MALANG  
2018**

## PENGESAHAN

RANCANG BANGUN 3D ORIENTED MOUSE CONTROLLER MENGGUNAKAN FINITE  
STATE MACHINE BERBASIS MIKROKONTROLLER ARDUINO

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Teknik

Disusun Oleh :  
Firdy Yantama  
NIM: 145150300111049

Skripsi ini telah diuji dan dinyatakan lulus pada  
28 Desember 2018  
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Dahnial Syaury, S.T., M.T., M.Sc  
NIK: 2016078704231002

Mochammad Hannats Hanafi Ichsan, S.ST, M.T  
NIK: 201405 881229 1 001

Mengetahui  
Ketua Jurusan Teknik Informatika



Tri Astoto Kurniawan, S.T., M.T., Ph.D  
NIK: 19710518 200312 1 001

A



### PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 29 Desember 2018



Firdy Yantama

NIM: 145150300111049



## KATA PENGANTAR

Puji syukur kehadiran Tuhan Yang Maha Esa, karena atas berkat dan hidayah-Nya, penulis dapat menyelesaikan dan menyusun laporan skripsi yang berjudul “RANCANG BANGUN 3D ORIENTED MOUSE CONTROLLER MENGGUNAKAN FINITE STATE MACHINE BERBASIS MIKROKONTROLLER ARDUINO”. Peneliti menyadari bahwa dalam penyusunan dan penyelesaian laporan skripsi ini tidak akan bisa selesai tanpa mendapatkan bantuan dari pihak-pihak terkait, oleh karena itu peneliti ingin memberikan rasa hormat serta ucapan terima kasih kepada :

1. Ibu Sri Wahyuni dan bapak Anang Kohar selaku Kedua orang tua penulis yang sudah memberikan dukungan baik dengan do’a maupun dukungan moral.
2. Bapak Dahnial Syauqy, S.T., M.T., M.Sc. dan Mochammad Hannats Hanafi Ichsan, S.ST, M.T selaku dosen pembimbing yang sudah mau meluangkan waktu, pikiran dan tenaganya dalam membimbing dan mengarahkan peneliti sehingga skripsi ini dapat diselesaikan.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika.
4. Seluruh teman-teman Teknik Komputer terutama yang sering berkumpul pada gedung C, Gehenna Novissima selaku teman terbaik penulis yang selalu memberi motivasi hidup dan mengajak berpikir dari sudut pandang berbeda, Cipto Bagus Jati Kusumo sebagai dosen pembimbing ketiga non-formal, Naufal Faiz Ramadhan, Irfan Pratomo Putra, Putra Wijaya, Agra Firmansyah, Aprilo Paskalis Polii, Bramadika Fardyan, dan kedua saudara penulis yaitu Farhan Dwiyan Putra (Sheccid) dan Favian Triyanda Naufal yang telah memberi motivasi kepada penulis untuk dapat memberi contoh yang baik sebagai anak pertama.
5. Seluruh civitas akademik Fakultas Ilmu Komputer Universitas Brawijaya yang telah membantu peneliti selama menempuh studi di Universitas Brawijaya

Dalam pengerjaan skripsi ini, penulis menyadari bahwa skripsi yang dikerjakan belum sempurna , sehingga untuk segala saran dan kritik akan sangat diterima oleh peneliti. Akhir kata, penulis harap skripsi ini dapat bermanfaat bagi orang yang menggunakannya.

Malang, 29 Desember 2018

Penulis  
nekloire@gmail.com

## ABSTRAK

Saat ini, perkembangan teknologi pada bidang *game* yang begitu pesat telah menciptakan istilah baru yang disebut *esports* ( *Electronic Sport* ) yang merupakan bidang olahraga elektronik. Hal ini memicu banyaknya bermunculan atlet profesional pada bidang *esports*. Salah satu *genre* favorit pada *esports* adalah *game first-person shooter* (FPS). Para atlet *esports* dapat menghabiskan waktu 8 hingga 14 jam sehari untuk berlatih. Sedangkan pemain *game* yang hanya sekedar hobi dapat menghabiskan waktu 2 hingga 6 jam. Banyak waktu yang dihabiskan untuk duduk diam di depan computer sehingga kurangnya aktifitas olahraga fisik dapat menimbulkan ancaman kesehatan seperti *Carpal Tunnel Syndrome*, cedera tangan, dan paru-paru melemah. Saat ini banyak dikembangkan *controller game* khusus yang bertujuan membuat penggunaanya lebih aktif melakukan pergerakan fisik. Dengan dibuatnya sistem *controller* khusus pada *game FPS* yang bentuknya dan cara penggunaannya menyerupai senjata api sungguhan, pemain *game* akan memiliki pergerakan fisik yang lebih aktif. Selain itu juga akan memberikan pengalaman yang menarik terhadap penggunaanya. Sistem dibuat dengan Arduino Uno yang menggunakan sensor *accelerometer* dan *gyroscope* yang memiliki *3D vector* sebagai input. Arah sudut dari perhitungan pitch dan dari kedua sensor tersebut akan mengendalikan arah *mouse*. *Complementary filter* digunakan untuk menghasilkan nilai sensor yang lebih akurat. Sistem menggunakan *Finite State Machine* untuk *state* utama dan *sub-state* dalam fungsionalitas sistem. Dari hasil pengujian yang dilakukan didapatkan nilai stabilitas sensor memiliki *error* 0,861% sedangkan *error* akurasi 1,61%. tingkat kepuasan pengguna berdasarkan standar *user experience* (UX) memberikan hasil *Useful* 76,64%, *Accessible* 76,38%, *Usable* 92,29%, dan *Desirable* 78,05%.

Kata kunci : *game*, *user experience* (UX), *finite state machine*, *complementary filter*, *3D vector*, *controller*

## ABSTRACT

Nowadays, rapid developing technology on game created a new term, Electronic Sport (esports). Many newcomer esports athletes rise in number. One of the favorite category on esports is first-person shooter (FPS). These esports athletes can spend up to 8 to 14 hours playing daily. While players whom play game as hobby can spend up to 2 to 6 hours daily. A lot of time spent by sitting still on front of computer makes their body lack physical movement, this causes health threats such as Carpal Tunnel Syndrome, hand injury, and weakened lungs. Therefore, a lot of custom controllers for games have been developed in order to give more body activities. Using a custom controller system with physical appearance and functions like a real gun in FPS game, its player will have more body activity. This system gives the user a new experience in playing FPS games. The system is using Arduino Uno with accelerometer and gyroscope as the pointer of 3D vector as input. Angle pointer from pitch and yaw calculation will determine the mouse movement. Complementary Filter is used in order to give more accuracy to the value generated by the sensors. The system will be implementing Finite-State Machine as main state and sub-state are needed for the system. Based on test to measure how stable and accurate the sensor, give error rate 0.861% and 1.61% for stable and accuracy respectively. The satisfaction results of its users based on User Experience (UX) standards are Useful 76.64%, Accessible 76.38%, Usable 92.29%, and Desirable 78.05%.

Keywords : game, user experience (UX), pitch, yaw, 3D vector, controller

## DAFTAR ISI

PENGESAHAN .....	ii
PERNYATAAN ORISINALITAS .....	iii
KATA PENGANTAR.....	iv
ABSTRAK.....	v
ABSTRACT .....	vi
DAFTAR ISI.....	vii
DAFTAR GAMBAR.....	x
DAFTAR TABEL.....	xii
BAB 1 PENDAHULUAN.....	1
1.1 Latar belakang.....	1
1.2 Rumusan masalah.....	3
1.3 Tujuan .....	3
1.4 Manfaat .....	3
1.5 Batasan Masalah.....	3
1.6 Sistematika Pembahasan.....	4
BAB 2 LANDASAN KEPUSTAKAAN .....	6
2.1 Tinjauan Pustaka.....	6
2.2 Dasar Teori.....	7
2.2.1 Finite State Machine .....	7
2.2.2 First-Person Shooter (FPS) .....	8
2.2.3 User Experience .....	8
2.2.4 Kuesioner .....	9
2.2.5 Mikrokontroler Arduino .....	9
2.2.6 I2C .....	12
2.2.7 Komunikasi Serial .....	13
2.2.8 Complementary Filter .....	13
2.2.9 Sensor MPU 6050.....	14
2.2.10 Sensor Joystick .....	16
BAB 3 METODOLOGI .....	18
3.1 Studi Literatur .....	18

3.2 Rekayasa Kebutuhan .....	18
3.3 Perancangan .....	19
3.4 Implementasi .....	19
3.5 Pengujian .....	19
3.6 Pengambilan Kesimpulan .....	19
<b>BAB 4 REKAYASA KEBUTUHAN.....</b>	<b>20</b>
4.1 Deskripsi Umum Sistem .....	20
4.1.1 Sampel Game Yang Digunakan .....	20
4.1.2 Ruang Lingkup Sistem .....	21
4.1.3 Batasan Sistem .....	21
4.2 Rekayasa Kebutuhan .....	21
4.2.1 Kebutuhan Fungsional .....	21
4.2.2 Kebutuhan Non-Fungsional .....	22
<b>BAB 5 PERANCANGAN DAN IMPLEMENTASI .....</b>	<b>24</b>
5.1 Perancangan Hardware .....	24
5.1.1 Perancangan Case Fisik Sistem .....	24
5.1.2 Perancangan Mikrokontroler Arduino Uno .....	26
5.2 Perancangan Software.....	30
5.2.1 Perancangan Software Pada Sisi Arduino Uno .....	30
5.2.2 Perancangan Flow Dari State .....	31
5.2.3 Perancangan Antarmuka.....	32
5.2.4 Perancangan Komunikasi Serial .....	32
5.2.5 Perancangan Orientasi Sensor MPU6050 .....	33
5.2.6 Perancangan Mouse Control .....	34
5.2.7 Perancangan W,A,S,D Movement dan Mouse Click .....	35
5.3 Implementasi Hardware .....	36
5.3.1 Implementasi Case Fisik Dan Mikrokontroler Arduino Uno .....	36
5.4 Implementasi Software .....	36
5.4.1 Implementasi Software Pada Sisi Arduino Uno .....	36
5.4.2 Implementasi Flow State Sistem.....	40
5.4.3 Implementasi Antarmuka Software .....	44
5.4.4 Implementasi Komunikasi Serial .....	44



5.4.5 Implementasi Mouse Control .....	46
5.4.6 Implementasi Orientasi MPU-6050 .....	48
5.4.7 Implementasi W,A,S,D Movement dan Mouse Click .....	49
BAB 6 PENGUJIAN .....	53
6.1 Pengujian Kestabilan Dan Akurasi Sensor MPU-6050 .....	53
6.1.1 Tujuan .....	53
6.1.2 Prosedur Pengujian .....	53
6.1.3 Hasil dan Analisis .....	54
6.2 Pengujian Fungsionalitas <i>Keyboard Event</i> , <i>Mouse Event</i> dan <i>Engine</i> Dari <i>Game</i> .....	59
6.2.1 Tujuan .....	59
6.2.2 Prosedur Pengujian .....	60
6.2.3 Hasil Dan Analisis .....	60
6.3 Pengujian <i>User Experience</i> (UX) Terhadap Pengguna Sistem .....	64
6.3.1 Tujuan .....	64
6.3.2 Prosedur Pengujian .....	64
6.3.3 Hasil Dan Analisis .....	65
BAB 7 PENUTUP .....	68
7.1 Kesimpulan .....	68
7.2 Saran .....	69
DAFTAR PUSTAKA .....	70
Lampiran A : Hasil Kuesioner .....	72



## DAFTAR GAMBAR

Gambar 2.1 Contoh State Machine .....	8
Gambar 2.2 Mikrokontroller Arduino Uno .....	10
Gambar 2.3 Layout Arduino Uno R3 .....	11
Gambar 2.4 Timing Komunikasi pada I2C .....	12
Gambar 2.5 Sensor MPU 6050.....	14
Gambar 2.6 <i>Accelerometer Piezoelektrik</i> .....	15
Gambar 2.7 <i>Gyroscope Piezoelektrik</i> .....	16
Gambar 2.8 Sensor Joystick .....	17
Gambar 3.1 Blok Diagram Metodologi .....	18
Gambar 3.2 Blok Diagram Sistem .....	19
Gambar 5.1 Blok Diagram Sistem .....	24
Gambar 5.2 Rancangan Dasar Case Fisik Sistem .....	25
Gambar 5.3 Tampak Atas Bagian Tengah Case Fisik.....	25
Gambar 5.4 Rancangan Hardware Secara Garis Besar .....	26
Gambar 5.5 Skema Sensor Joystick dan Mikrokontroller Arduino .....	28
Gambar 5.6 Koneksi Sensor MPU-6050 Pada Mikrokontroller Arduino .....	29
Gambar 5.7 Rangkaian Pull Down Resistor.....	29
Gambar 5.8 Hubungan Button Dengan Mikrokontroller Arduino .....	30
Gambar 5.9 Flowchart Pada Software Arduino .....	30
Gambar 5.10 Flow State Sistem .....	31
Gambar 5.11 Flowchart Pembacaan Data Serial .....	33
Gambar 5.12 Ilustrasi Orientasi Sensor .....	33
Gambar 5.13 Flowchart Mouse Control .....	34
Gambar 5.14 mplementasi <i>Case Fisik 3D Vector Mouse Controller</i> .....	36
Gambar 5.15 Antarmuka Software .....	44
Gambar 5.16 Data String Dari Mikrokontroller Arduino .....	45
Gambar 6.1 Hasil Pengujian Stabilitas Nilai .....	55
Gambar 6.2 Posisi Alat Saat Berdiri Tegak.....	55
Gambar 6.3 Nilai Awal Saat Alat Dinyalakan .....	56
Gambar 6.4 Alat Diputar 90 Derajat Ke Kiri .....	56

Gambar 6.5 Alat Diputar 90 Derajat ke Kanan (Sudut Awal).....	56
Gambar 6.6 Alat Diputar 90 Derajat ke Kanan dari sudut 0 derajat.....	56
Gambar 6.7 Alat diputar 90 derajat ke kanan dari sudut -90 derajat .....	56
Gambar 6.8 Message Event Keyboard Sungguhan .....	61
Gambar 6.9 Message Event Alat.....	61
Gambar 6.10 Event Yang Dihasilkan Mouse Sungguhan .....	62
Gambar 6.11 Event Mouse Dari Alat .....	62
Gambar 6.12 Pengujian Alat Pada Game Minecraft.....	62
Gambar 6.13 Pengujian Alat Pada Game Counter Strike 1.6.....	63
Gambar 6.14 Responden Sedang Mencoba Alat .....	65
Gambar 6.15 Chart Hasil Kuesioner .....	66



## DAFTAR TABEL

Tabel 2.1 Perbedaan Dengan Penelitian Sebelumnya .....	6
Tabel 2.2 Spesifikasi Mikrokontroller Arduino Uno .....	10
Tabel 2.3 Spesifikasi Pin Arduino Uno .....	11
Tabel 2.4 Spesifikasi Pin Sensor MPU-6050 .....	15
Tabel 2.5 Spesifikasi Pin Sensor Joystick .....	17
Tabel 4.1 Kebutuhan Perangkat Keras Sistem .....	22
Tabel 4.2 Kebutuhan Perangkat Lunak Sistem .....	23
Tabel 5.1 Pin Yang Digunakan .....	26
Tabel 5.2 Pin Sada Sensor Joystick .....	28
Tabel 5.3 Kolom Antarmuka Software .....	32
Tabel 5.4 Pemicu Perintah Keyboard dan Mouse Click .....	35
Tabel 5.5 Source Code Software Mikrokontroller Arduino .....	37
Tabel 5.6 Initiation State .....	40
Tabel 5.7 Standby State .....	41
Tabel 5.8 Oriented, Left Rotation, Right Rotation State .....	42
Tabel 5.9 Pemicu Oriented, Left Rotation, Right Rotation State .....	42
Tabel 5.10 Konfigurasi Komunikasi Serial .....	44
Tabel 5.11 Tag Pada String .....	45
Tabel 5.12 Fungsi Pembaca Akhir Tag .....	46
Tabel 5.13 Penanganan Mouse Pada Software .....	46
Tabel 5.14 Pergerakan Mouse sesuai State .....	48
Tabel 5.15 State Pada Software .....	49
Tabel 5.16 Source Code Keyboard .....	50
Tabel 5.17 Source Code Mouse Click .....	51
Tabel 6.1 Hasil Pengujian Stabilitas Sensor MPU-6050 .....	57
Tabel 6.2 Hasil Pengujian Akurasi Arah Sudut Yaw .....	58
Tabel 6.3 Penjelasan Message Event Keyboard .....	60
Tabel 6.4 Hasil Pengujian Fungsionalitas Alat .....	63

## BAB 1 PENDAHULUAN

Pada bab ini dijelaskan awal mula latar belakang penelitian yang tentunya akan menghasilkan beberapa rumusan masalah. Rumusan masalah tersebut akan memberikan tujuan pada penelitian, yaitu menjawab rumusan masalah yang ditemukan. Selain itu akan dibahas juga mengenai manfaat, batasan masalah, serta sistematika pembahasan dari penelitian ini.

### 1.1 Latar belakang

Perkembangan teknologi komputer yang selalu terjadi seiring waktu perlahan lahan telah membentuk ulang fungsi dari suatu komputer. Komputer yang dulunya hanya mengerjakan komputasi secara literal kini juga memiliki fungsi lain seperti sebagai *entertainment* dimana banyak dikembangkan game – game berbasis komputer. Perkembangan ini diperlukan tidak semata hanya untuk hiburan, namun juga pembelajaran sebagai simulasi penerapan logika dan fisik untuk diimplementasikan ke dalam sebuah *game*. Selain itu, kemajuan teknologi pada game komputer telah menciptakan sebuah istilah baru yang disebut *esports*.

*Esports*, merupakan singkatan dari *electronic sport*, dimana merupakan kegiatan adu ketangkasan secara fisik ataupun non-fisik yang dilakukan oleh individu atau kelompok, dengan menggunakan perangkat elektronik yaitu video game. Ada tiga acuan utama untuk sebuah permainan atau hiburan untuk dapat dikategorikan menjadi menjadi olahraga: bermain, bersaing dan melihat. Dari kompetisi video game skala besar pertama di tahun 1970an sampai sekarang, *esports* telah mengalami lintasan yang mirip dengan olahraga *offline*. Memang, kemampuan bermain dan bersaing merupakan langkah penting dalam transformasi dari *game* ke *sport*. Namun, penyiaran dan penayangan adalah komponen penting untuk memungkinkan adopsi dan popularitas yang meluas. Sama seperti dengan olahraga sungguhan, *esports* membutuhkan unsur-unsur ini (Paradise, The importance of streaming to e-sports, 2017) .

Pada tahun 2014, Twitch Amazon mencatat 100 juta penonton unik per bulan, menghasilkan peningkatan penayangan sebesar 66 persen (dari 60 juta di tahun 2013). Pada tahun 2015, Twitch mengalirkan lebih dari 241 miliar menit, atau sekitar 460.000 tahun konten. Keterlibatan pada layanan streaming populer sangat mengesankan, dengan rata-rata penampil Twitch melakukan penayangan lebih dari 421,6 menit per bulan. Sebagai penonton, pemirsa *esports* sangat terlibat, dan total penonton diperkirakan akan tumbuh menjadi 345 juta pada 2019, menurut Newzoo ( Techcrunch.com, 2017).

Salah satu dari kategori *esports* yang populer adalah *FPS (First Person Shooter)*, yaitu *game* berbasis tembak-menembak menggunakan senjata api atau alat lainnya yang menampilkan lingkungan game dari sudut pandang pemain. Dari statistik yang dikumpulkan oleh Stista.com mengenai *esports* didapatkan informasi bahwa peminat *esports* baik jumlah pemain profesional atau hanya sekedar penonton kegiatan *esports*, pada tahun 2016 terdapat 161 juta individu.

Selain itu jumlah nominal uang yang dikeluarkan secara global dalam pertandingan *esports* pada tahun 2016 saja bertotal 492 juta USD (Stista.com, 2018).

Walaupun termasuk dalam kategori olahraga, *esports* memiliki beberapa ancaman kesehatan pada atletnya. Beberapa diantaranya yaitu *Carpal Tunnel Syndrome* dan cedera tangan akibat pergerakan yang terlalu berulang-ulang saat tangan menggerakkan *mouse* di meja komputer. Paru-paru yang melemah karena postur duduk yang salah dan tidak aktifnya anggota badan dalam jangka waktu yang lama, hal ini dialami oleh beberapa atlet *esports* yang terlalu lama duduk di depan komputer. Melemahnya reflek yang dipicu oleh indra selain penglihatan dan pendengaran, karena pada *esports* reflek hanya dipicu oleh suara dari *game* dan grafis yang ditampilkan *game* tersebut (Chua, 2017). Karena itu untuk meningkatkan keaktifan gerak raga dari atlet *esports*, saat ini telah banyak dikembangkan perangkat khusus untuk *game esports* seperti *controller* Nintendo Wii dalam bermain *game* bulu tangkis dimana pemain harus menggerakkan *controller* tersebut layaknya bermain bulu tangkis sungguhan. Virtual Reality merupakan contoh yang paling mencolok dalam pengembangan *controller esports* dimana mulai diimplementasikan *controller* yang mengimitasi gerakan tubuh manusia sungguhan dalam pengendaliannya.

Oleh karena itu, penulis tertarik menciptakan *controller* berbentuk senjata api yang dirancang khusus untuk *game* berbasis FPS, dengan tujuan memberikan *esports* aktifitas fisik yang aktif layaknya olahraga yang sesungguhnya demi mengurangi resiko gangguan kesehatan pada atlet *esports*. *Controller* ini berbasis mikrokontroler Arduino Uno yang akan menggunakan *axis 3d* dari output sensor MPU-6050 sebagai pengendali *cursor* dimana penggunaanya harus mengarahkan perangkat ini ke sudut yang sesuai. Output dari sensor MPU-6050 akan dikonversi menjadi arah sudut *pitch* dan *yaw* menggunakan *complementary filter*. Perangkat ini memiliki 2 *button* yang terletak pada area pelatuk pistol sebagai pengganti fungsi klik kiri dan klik kanan pada *mouse* konvensional, serta *joystick* pada bagian belakang untuk melakukan pergerakan pada *game* berbasis *First-Person*. Dalam perancangan *software* dari sistem ini menggunakan *Finite State Machine* untuk memenuhi beberapa kondisi pada sistem dalam memudahkan untuk bermain *game* FPS tanpa *mouse* konvensional seperti kondisi saat pemain ingin memutar arah sudut pandang dari *game* berbasis FPS dan saat melakukan pergerakan *cursor* pada sisi arah depan saja. Pada *mouse* konvensional pergerakan tersebut dapat dilakukan dengan mudah, namun pada sistem ini akan mengakibatkan pengguna membelakangi layar komputer saat ingin melakukan perputaran kearah belakang pada *game*, oleh karena itu digunakan *Finite State Machine* untuk mengatasi permasalahan ini. Alat akan diuji untuk mengukur *user experience* (UX) yang bertujuan untuk mengetahui kenyamanan penggunaan sistem di tangan *user*. Dengan adanya perangkat ini diharapkan dapat membuat *esports* menjadi olahraga yang aktif menggerakkan anggota tubuh para atlet *esports*.



## 1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan, maka dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana cara membuat *controller* yang memiliki axis 3 dimensi dan mengolah datanya menjadi suatu titik arah di dalam game?
2. Bagaimana cara berkomunikasi dan menerjemahkan data yang dikirim Mikrokontroler Arduino untuk menghasilkan output yang sesuai pada PC?
3. Bagaimana performa kinerja sistem yang telah dirancang ?

## 1.3 Tujuan

Adapun tujuan dilakukannya penelitian berikut adalah :

1. Melakukan kalkulasi pada 3 axis yang di dapat dari sensor sistem untuk mendapatkan suatu nilai arah.
2. Membuat algoritma yang akurat dalam memetakan koordinat 3 axis dari sistem yang dibuat ke dalam input 2 axis komputer
3. Untuk menguji performa kinerja sistem yang dibuat ketika digunakan.

## 1.4 Manfaat

Manfaat yang dihasilkan dari penelitian ini adalah sebagai berikut :

1. Pemain *game esports* pada genre FPS, dapat merasakan pengalaman seperti keadaan yang sesungguhnya dalam mengendalikan senjata api pada dunia nyata .
2. Atlit *esports*, akan lebih banyak melakukan aktifitas pergerakan tubuh demi menjaga kesehatannya.

## 1.5 Batasan Masalah

Batasan masalah merupakan batasan-batasan yang terdapat pada sistem yang diteliti. Beberapa batasan masalah pada penelitian ini adalah :

1. Control pada sistem yang dibuat hanya untuk menggantikan fungsi *controller mouse* dan sebagian fungsi *keyboard* saja, dimana hanya mempunyai fungsi *left-click*, *right-click*, dan pengubahan koordinat x,y pada desktop, serta tombol *keyboard W,A,S,D*.
2. Pergerakan yang dihasilkan oleh *controller* yang dibuat tidak selaras secara benar-benar akurat pada output yang dihasilkan pada komputer. Semisal pengguna kontroller berputar 360 derajat, maka visualisasi pergerakan cursor pada game yang diuji tidak harus berputar tepat 360 derajat karena terdapat beberapa faktor yang mempengaruhi keakuratan pergerakan cursor

diantaranya adalah konfigurasi *mouse sensitivity* pada *operating system* yang digunakan dan game yang menjadi lingkungan pengujian. Pergerakan controller hanya selaras dengan konfigurasi tertentu yang telah ditetapkan dalam game.

3. Sistem diujikan menggunakan metode *State Machine*.
4. Sistem diujikan pada game berbasis FPS yaitu Minecraft dan Counter-Strike 1.6 pada *operating system* berbasis WINDOWS

## 1.6 Sistematika Pembahasan

Sistematika pembahasan penyusunan laporan Skripsi bertujuan untuk memberikan gambaran dan uraian dari beberapa bab yang terdapat pada laporan ini, sebagai berikut :

### Bab 1 PENDAHULUAN

Terdiri dari atas Latar Belakang Masalah yang merupakan sumber utama ide untuk melakukan penelitian, Rumusan Masalah, Tujuan, Manfaat, Batasan Masalah dan Sistematika Pembahasan.

### Bab 2 LANDASAN KEPUSTAKAAN

Terdiri acuan referensi yang memiliki relevansi terhadap penelitian yang dilakukan seperti spesifikasi dari *hardware* sistem yang menggunakan mikrokontroler Arduino Uno, Sensor MPU-6050 dan juga metode yang digunakan seperti *complementary filter*, Finite State Machine dan pengukuran nilai kepuasan pengguna berdasarkan *user experience* (UX).

### Bab 3 METODOLOGI

Terdiri tahapan penelitian yang dilakukan dalam pengembangan sistem. Dimulai dari studi literatur, rekayasa kebutuhan yang diperlukan sistem, perancangan sistem secara *hardware* dan *software*, implementasi sistem yang telah dirancang, pengujian performa sistem, dan diakhiri dengan pengambilan kesimpulan.

### Bab 4 REKAYASA KEBUTUHAN

Terdiri dari rekayasa kebutuhan dari sistem yang akan dibuat. pada bab ini dijelaskan kebutuhan fungsional yang harus dipenuhi sistem, kebutuhan non-fungsional serta kebutuhan hardware dan software dari sistem yang akan dibuat.

### Bab 5 PERANCANGAN DAN IMPLEMENTASI

Terdiri dari perancangan dan implementasi sistem dari sisi software dan hardware. Bab ini menjelaskan alur dan diagram blok dari sistem yang dibuat. *Hardware* menggunakan mikrokontroler



Arduino Uno yang dimuat dalam *case* fisik berbahan akrilik. *Software* dibuat menggunakan Bahasa C untuk program pada Arduino Uno dan Visual Basic untuk program yang akan dijalankan pada PC.

## **Bab 6            PENGUJIAN**

Merupakan bab yang membahas pengujian dari sistem yang telah dirancang. Pengujian sistem ini dilakukan dengan cara meminta pengguna untuk mencoba menggunakan sistem dan mengisi kuesioner yang dirancang untuk mengukur *user experience* (UX).

## **Bab 7            PENUTUP**

Berisi kesimpulan dari keseluruhan pembuatan sistem mulai dari perancangan, implementasi, dan pengujian. Pada bab ini diuraikan jawaban yang terdapat pada rumusan masalah dan hasil yang didapatkan dari pengujian sistem.



## BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini akan dibahas mengenai poin tinjauan pustaka berupa jurnal-jurnal referensi yang menjadi bahan acuan dan poin dasar teori yang mencakup pengetahuan dan istilah-istilah teknis yang terdapat pada penelitian.

### 2.1 Tinjauan Pustaka

Dalam penulisan laporan skripsi ini, peneliti telah menggali informasi mengenai beberapa penelitian yang telah dilakukan sebelumnya yang memiliki relevansi sebagai bahan acuan pada penelitian ini. Jurnal Rishab Kaw dari Shri Mata Vaishno Devi University, India yang berjudul *"Design of Tilt Sensing Mouse Glove Device using Arduino Uno"* yang membahas tentang mouse berbentuk sarung tangan yang mengenali gestur tangan sebagai input dan mengubahnya menjadi sinyal computer menggunakan algoritma matematis. Perangkat mouse dalam jurnal ini berbasis *3-axis accelerometer*.

Jurnal A. Heydari Gorji dan Safavi dari Electrical Engineering and Computer Science, University of California yang berjudul *"Head-mouse: A simple cursor controller based on optical measurement of head tilt"*. Jurnal ini membahas tentang perangkat *mouse-controller* yang dikenakan pada kepala dan mengendalikan cursor mouse menggunakan kemiringan dari kepala sebagai penggerak *mouse*. Data dari perangkat ini dikirim secara wireless menggunakan Bluetooth Low Energy (BLE).

Jurnal dari Sohail Ahmed ( Muffakham Jah College of Engineering and Technology ), Mohammed Abdullah Zubair ( IIT Hyderabad ), Irshad Basha Shaik ( RITS ) yang berjudul *"Accelerometer Based Wireless Air Mouse Using Arduino Micro-Controller Board"* Jurnal ini membahas tentang implementasi mikrokontroler Arduino menggunakan sensor *accelerometer* sebagai pengganti mouse. Alat berbentuk menyerupai konsol Playstation sebagai pengganti *mouse* konvensional.

Berdasarkan jurnal yang telah disebut diatas, maka telah diketahui bahwa penelitian tentang *mouse-controller* berbentuk khusus dalam menggantikan *mouse* konvensional telah dilakukan sebelumnya. Untuk perbedaan pada penelitian yang pernah dilakukan sebelumnya terhadap penelitian penulis dapat dilihat pada Tabel 2.1 .

**Tabel 2.1 Perbedaan Dengan Penelitian Sebelumnya**

No	Penulis, Judul sebelumnya	Penelitian Sebelumnya	Perbedaan
1	Rishab Kaw, <i>Design of Tilt Sensing Mouse Glove Device using Arduino Uno</i>	Perangkat mouse berbentuk saru tangan yang menerima input mouse dari gestur tangan.	Sistem menggunakan model senjata api. Input berupa pergerakan sistem tersebut serta beberapa tombol mouse yang disediakan

2	A. Heydari Gorji dan Safavi, <i>Head-mouse: A simple cursor controller based on optical measurement of head tilt</i>	Perangkat mouse yang dikenakan pada kepala dan inputnya diambil dari nilai kemiringan kepala untuk orang yang memiliki disabilitas.	Perangkat ini dirancang khusus untuk <i>game</i> berbasis FPS
3	Sohel, Mohammed Abdullah Zubair, dan Irshad Basha Shaik, <i>Accelerometer Based Wireless Air Mouse Using Arduino Micro-Controller Board</i>	Menggunakan perangkat mouse berbentuk konsol Playstation berbasis sensor <i>wireless</i> Bluetooth	Perangkat berbentuk senjata api dan data yang dikirim dari sistem kepada PC menggunakan Komunikasi Serial melalui kabel fisik.

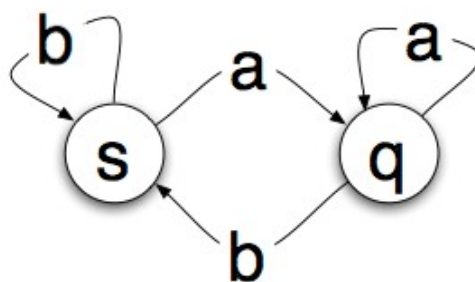
Ketiga penelitian tersebut memiliki metode pengolahan data dan cara berkomunikasi yang berbeda dalam membuat sistem dalam bidang yang sama yaitu sebagai pengganti *mouse* konvensional. Sensor yang digunakan pada penelitian yang telah dilakukan tersebut menggunakan *gyroscope*, *accelerometer*, dan *joystick* dalam menggerakan *cursor* dari *mouse*.

## 2.2 Dasar Teori

Dalam Sub-bab ini diberikan penjelasan mengenai teori-teori yang digunakan pada penelitian ini seperti, Algoritma *Finite State Machine*, Sensor *Accelerometer* dan *Gyro* MPU 6050, Mikrokontroler Arduino, *User Experience* (UX), *complementary filter*, dan I2C.

### 2.2.1 Finite State Machine

Finite State Machine ditemukan oleh Edward Forrest Moore (1925-2003), professor matematika dan komputer sains dari Amerika. Finite State Machine adalah suatu kondisi matematis yang digunakan untuk mendesain suatu algoritma. Suatu input berbeda yang telah didefinisikan akan dibaca dan memberikan perpindahan pada suatu *state* atau kondisi. Setiap *state* yang telah dirancang memiliki fungsi dan tujuan yang berbeda dalam menjalankan prosesnya. Penggambaran analogi dari Finite State Machine dapat dilihat pada Gambar 2.1.



Gambar 2.1 Contoh State Machine

Seperti pada Gambar 2.1, huruf dalam lingkaran merupakan suatu nama *state*. Sedangkan huruf a dan b pada garis panah menunjukan suatu kondisi yang akan memicu perpindahan *state*.

### 2.2.2 First-Person Shooter (FPS)

*First-person shooter* adalah jenis dari *game* komputer 3 dimensi yang menampilkan visualiasi lingkungan game dengan sudut pandang orang pertama. Tidak seperti game secara umumnya dimana sudut pandang yang digunakan adalah orang ketiga yaitu tampilan visual menampilkan lingkungan game dari belakang karakter yang dikendalikan dalam game. Elemen utama dari jenis *game* ini adalah mekanisme bertarung terutama menggunakan senjata api.

### 2.2.3 User Experience

Berdasarkan dokumen publikasi dari ISO (*the International Organization for Standaritation*) dengan nomor referensi ISO 9241-210, *User Experience* (UX) adalah persepsi seseorang dan responnya dari penggunaan sebuah produk, sistem, atau jasa. *User Experience* adalah suatu ilmu yang memiliki focus untuk mempelajari lebih dalam tentang pengguna suatu produk atau jasa dalam mengukur nilai kepuasan. Yang dipelajari meliputi kebutuhan, nilai, kemampuan, dan Batasan dari pengguna. Ilmu ini bertujuan untuk mengembangkan interaksi yang baik antara pengguna dengan produk atau jasa yang disediakan. Beberapa faktor yang mempengaruhi interaksi dengan pengguna berdasarkan model Peter Morville adalah :

1. *Useful*, konten dari produk atau jasa berguna dan dapat memenuhi suatu kebutuhan.
2. *Usable*, produk atau jasa harus mudah digunakan oleh pengguna.
3. *Desirable*, produk atau jasa harus dapat mempengaruhi emosi dari pengguna sehingga pengguna akan senang dan produk atau jasa tersebut sangat diinginkan oleh pengguna.
4. *Findable*, suatu produk atau jasa harus mudah ditemukan atau mudah diakses oleh suatu pengguna dari berbagai lokasi.

5. *Accessible*, produk atau jasa bisa digunakan oleh berbagai kalangan termasuk pengguna yang memiliki disabilitas atau batasan tertentu.
6. *Credible*, pada pengguna harus tumbuh rasa percaya pada suatu produk atau jasa yang disediakan.

#### 2.2.4 Kuesioner

Kuesioner merupakan suatu metode dalam mengumpulkan informasi dalam melakukan analisis terhadap suatu studi kasus yang sedang diteliti. Menurut definisi dari Kamus Besar Bahasa Indonesia (KBBI), kuesioner memiliki arti alat riset atau survei yang terdiri atas serangkaian pertanyaan tertulis, bertujuan mendapatkan tanggapan dari kelompok orang terpilih melalui wawancara pribadi atau melalui pos; daftar pertanyaan. Metode ini akan mengukur seberapa luas dan sentiment yang diekspresikan dalam suatu wawancara terhadap pemberi informasi yang disebut responden.

Terdapat 2 jenis pertanyaan pada kuesioner yaitu pertanyaan terbuka dan pertanyaan tertutup. Pertanyaan terbuka merupakan pertanyaan yang memberikan pilihan respon secara bebas. Respon dalam pertanyaan ini sulit diukur karena lebih susah dalam menterjemahkan nilai dari respon tersebut. Sedangkan pertanyaan tertutup adalah suatu pertanyaan yang memberi respon terbatas dalam suatu kuesioner. Pertanyaan tertutup pada umum menyediakan beberapa opsi pilihan respon dimana setiap opsi tersebut mempresentasikan suatu nilai yang dapat diukur.

#### 2.2.5 Mikrokontroler Arduino

Arduino merupakan mikro single-board yang bersifat open-source yang diturunkan dari Wiring platform dan dirancang dengan tujuan untuk memudahkan penggunaan elektronik pada berbagai bidang. Hardwarenya menggunakan prosesor Atmel AVR dan dapat deprogram sesuai keperluan menggunakan bahasa pemrograman sendiri.

Arduino juga merupakan sebuah platform hardware terbuka yang ditujukan kepada para pengembang yang ingin membuat purwarupa peralatan elektronik interaktif berdasarkan hardware dan software yang fleksibel dan mudah digunakan. Mikrokontroler diprogram menggunakan bahasa pemrograman arduino yang memiliki kemiripan syntax dengan bahasa pemrograman C.

Arduino menggunakan keluarga mikrokontroler dari ATmega yang dirilis oleh Atmel sebagai basis, namun beberapa instansi lain juga membuat clone arduino dengan menggunakan mikrokontroler lain dan tetap kompatibel dengan arduino pada level hardware. Untuk fleksibilitas, program dimasukkan melalui bootloader meskipun ada opsi untuk membypass bootloader dan menggunakan downloader untuk memprogram mikrokontroler secara langsung melalui port ISP. Bentuk board Arduino Uno dapat dilihat pada Gambar 2.2





**Gambar 2.2 Mikrokontroler Arduino Uno**

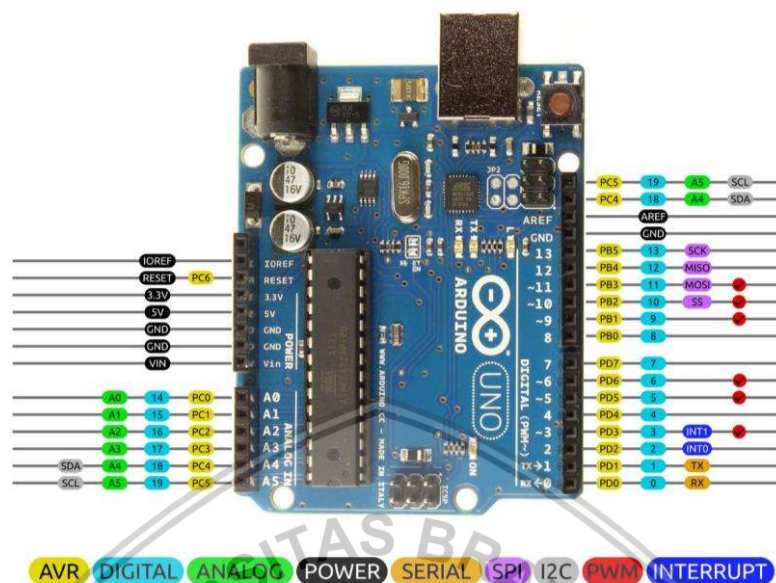
(sumber <https://store.arduino.cc/usa/arduino-uno-rev3>)

Spesifikasi dari Arduino Uno r3 dijelaskan pada Tabel 2.2. Arduino Uno R3 juga menambahkan pin SDA (*Serial Data*) dan SCL (*Serial Clock*) di samping AREF. Selain itu, ada dua pin baru yang ditempatkan di dekat pin RESET. Salah satunya adalah IOREF yang memungkinkan perisai untuk beradaptasi dengan tegangan yang disediakan dari papan. Yang lainnya adalah tidak terhubung dan dicadangkan untuk tujuan masa depan. Uno R3 bekerja dengan semua perisai yang ada tetapi dapat beradaptasi dengan perisai baru yang menggunakan pin tambahan ini.

**Tabel 2.2 Spesifikasi Mikrokontroler Arduino Uno**

Mikrokontroler	ATmega328P – 8 bit AVR family microcontroller
Voltase pengoprasian	5V
Voltase input yang disarankan	7V – 12V
Jumlah pin input analog	6 (A0 – A5)
Jumlah pin input digital	14
Arus DC pada input	40 mA
Flash Memory	32kb (0,5 kb digunakan sebagai <i>bootloader</i> )
SRAM	2kb
EEPROM	1kb
Frekuensi ( <i>clock speed</i> )	16 MHz

Skema pin pada board Arduino Uno R3 secara lebih detail dapat dilihat pada Gambar 2.3 Layout Arduino Uno R3 dan Tabel 2.3.



**Gambar 2.3 Layout Arduino Uno R3**

Arduino Uno R3 memiliki 14 pin digital, 5 pin analog, 2 pin ground, 2 pin power untuk 3,3 volt dan 5 volt. Arduino Uno R3 juga mendukung komunikasi I2C dengan tersedianya pin SCL dan SDA.

**Tabel 2.3 Spesifikasi Pin Arduino Uno**

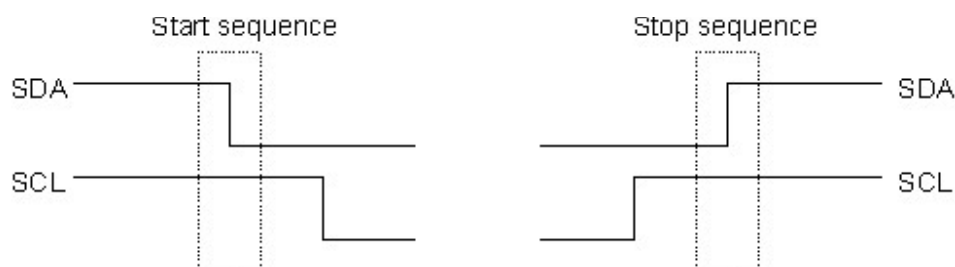
Kategori Pin	Nama Pin	Keterangan
Daya	Vin	Voltase <i>input</i> dari mikrokontroller jika menggunakan sumber tegangan secara eksternal.
	3,3V	Suplai voltase yang dihasilkan <i>on-board</i> sebagai sumber daya untuk komponen lain yang digunakan pada mikrokontroller seperti sensor. Memberikan tegangan 3,3V
	5V	Suplai voltase yang dihasilkan <i>on-board</i> sebagai sumber daya untuk komponen lain yang digunakan pada mikrokontroller seperti sensor. Memberikan tegangan 5V
	GND	Pin sebagai <i>ground</i> arus tegangan
Reset	Reset	Mengembalikan konfigurasi mikrokontroller ke keadaan semula.
Pin Analog (6 buah)	A0, A1, A2, A3, A4, A5	Digunakan untuk memberikan input analog mulai dari 0 hingga 5V



<i>Input / Output</i> digital (14 buah)	0 hingga 13	Dapat digunakan sebagai <i>input</i> maupun <i>output</i>
Serial	0(Rx), 1(Tx)	Digunakan untuk menerima transmit data serial
Interrupt	Pin 2,3	Digunakan untuk memicu <i>interrupt</i> pada mikrokontroler
PWM	3,5,6,9,11	Memberikan output PWM 8-bit
SPI	10 (SS), 11 (MOSI), 12 (MISO), dan 13 (SCK)	Digunakan untuk komunikasi SPI
LED	13	Pin digunakan untuk menyalakan LED
TWI	A4 (SDA), A5 (SCA)	Digunakan untuk komunikasi secara serial terhadap sensor
AREF	AREF	Menyediakan voltase referensi untuk sumber daya

### 2.2.6 I2C

Inter-integrated Circuit (I2C) Protocol adalah protokol yang ditujukan untuk memungkinkan beberapa sirkuit terintegrasi digital “slave” (“chip”) untuk berkomunikasi dengan satu atau lebih chip “master”. Seperti Serial Peripheral Interface (SPI), itu hanya ditujukan untuk komunikasi jarak pendek dalam satu perangkat. Seperti Asynchronous Serial Interfaces (seperti RS-232 atau UART), hanya membutuhkan dua kabel sinyal untuk bertukar informasi. Protokol ini memungkinkan untuk memberikan jalur komunikasi terpisah dan tidak mengganggu satu sama lain antara perangkat I2C lainnya walaupun menggunakan pin / port yang sama sekaligus. Sekuensi komunikasi I2C ditunjukkan pada Gambar 2.4.



Gambar 2.4 Timing Komunikasi pada I2C

Ketika perangkat *master* ingin berkomunikasi pada perangkat *slave* maka yang perangkat *master* akan melakukan *start sequence* yaitu nilai dari SDA berpindah

dari *high* ke *low* saat logika SCL sedang *high*. Sedangkan untuk berhenti melakukan komunikasi akan dilakukan *stop sequence* yaitu nilai dari SDA berpindah dari *low* ke *high* saat logika SCL sedang *high*. Saat data sedang dikirim, kondisi SDA harus stabil dan kondisi SCL harus *high*. setiap 8 bit data terkirim maka harus dikirimkan 1 sinyal *acknowledge* sebagai penanda bahwa data berhasil terkirim. Dari hal tersebut dapat diketahui bahwa terdapat 9 gelombang *clock* SCL setiap 8 bit data.

### 2.2.7 Komunikasi Serial

Dalam komputasi, *port* serial adalah suatu antarmuka dari komunikasi serial dimana suatu informasi dikirim masuk atau keluar 1 bit setiap waktu. Pada Arduino komunikasi serial digunakan untuk berkomunikasi antara suatu mikrokontroler Arduino dengan sebuah komputer atau perangkat lainnya. Semua jenis mikrokontroler Arduino setidaknya punya 1 buah *port* serial yang disebut UART atau USART. Komunikasi serial pada Arduino dapat dilakukan melalui pin 0(RX) dan pin 1 (TX) begitu juga dengan komputer menggunakan USB..

### 2.2.8 Complementary Filter

*Complementary Filter* merupakan suatu metode matematis yang menggabungkan sinyal yang bergerak lambat dengan sinyal yang bergerak cepat. (Robert Mahony, 2006). Biasanya *filter* ini digunakan untuk menggabungkan sinyal dari *accelerometer* yang bergerak lambat dengan sinyal dari *gyroscope* yang bergerak cepat. *Filter* ini memasukkan suatu sinyal ke dalam *low-pass filter* dan sinyal lainnya ke dalam *high-pass filter*. Rumus dari *low-pass filter* ini ditunjukkan pada persamaan 2.1 :

$$y[n] = (1 - \alpha) \cdot x[n] + \alpha \cdot y[n - 1] \quad 2.1$$

dimana nilai dari variable pada persamaan 2.1 adalah:

$x[n]$  = nilai *pitch / roll / yaw* dari *accelerometer*

$y[n]$  = hasil dari filter yang harus di *feedback* pada program

untuk cara menghitung *high-pass filter* didapatkan dari persamaan 2.2 :

$$y[n] = (1 - \alpha) \cdot y[n - 1] + (1 - \alpha)(x[n] - x[n - 1]) \quad 2.2$$

dimana nilai dari variable pada persamaan 2.2 adalah:

$x[n]$  = nilai *pitch / roll / yaw* dari *gyroscope*

$y[n]$  = hasil dari filter yang harus di *feedback* pada program

nilai variable  $n$  pada persamaan 2.1 dan persamaan 2.2 merupakan indikasi sampel.  $\alpha$  adalah suatu nilai constant waktu yang merupakan batas dimana

*accelerometer* berhenti dan diambil alih pembacaannya oleh *gyroscope* dan juga sebaliknya. Nilai *alpha* pada kedua persamaan tersebut harus sama. Nilai *alpha* berdasarkan persamaan tersebut biasanya diata 0,5. Nilai dari *alpha* adalah  $\alpha = (\tau) / (\tau + dt)$  dimana variable *tau* adalah constant waktu seberapa cepat respon pembacaan dan  $dt = 1/\text{frekuensi sampling}$ . Untuk mengimplementasikan *complementary filter* dapat digunakan persamaan :

$$\text{angle} = (1 - \alpha) * (\text{angle} + \text{gyro} * dt) + (\alpha) * (\text{acc}) \quad 2.3$$

Dari persamaan tersebut jika nilai hasil kalkulasi *alpha* rendah pada *accelerometer*, maka output yang dihasilkan dari *filter* tersebut akan lebih mempercayai sisi *gyroscope*.

### 2.2.9 Sensor MPU 6050

Sensor MPU-6050 adalah sebuah modul dengan *accelerometer* dan *gyroscope* berinti yang merupakan 6 axis Motion Processing Unit dengan penambahan regulator tegangan yang membuat modul ini mampu berjalan pada tegangan supply sebesar 3-5VDC. Modul ini memiliki interface I2C yang dapat disambungkan langsung ke MCU yang memiliki fasilitas I2C. Untuk mengakses sensor ini pada I2C dapat digunakan adress 0x69

Sensor ini memiliki fasilitas hardware internal 16 bit ADC untuk setiap kanalnya. Sensor ini akan menangkap nilai kanal axis X, Y dan Z bersamaan dalam satu waktu. Bentuk dari sensor ini dapat dilihat pada Gambar 2.5



Gambar 2.5 Sensor MPU 6050

(sumber [http://www.electrooobs.com/images/Robotica/tut\\_6/mpu6050.jpg](http://www.electrooobs.com/images/Robotica/tut_6/mpu6050.jpg))

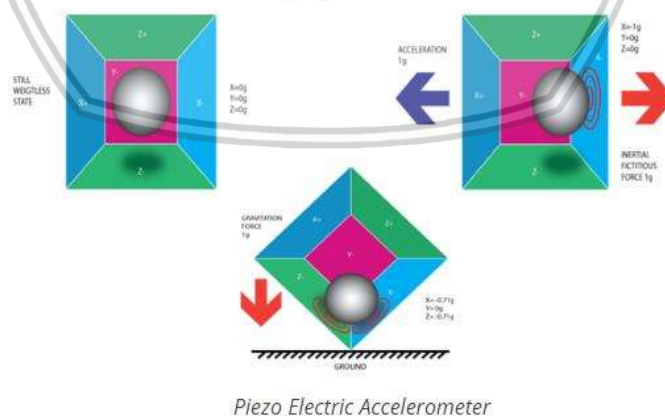
Sensor ini memiliki spesifikasi pin yang ditunjukkan pada Tabel 2.4 dimana terdapat 8 buah pin, 2 pin untuk arus tegangan (VCC dan *ground*), 4 pin sebagai pengiriman data (SCL, SDA, XCL, dan XDA), 1 buah pin *Interrupt*, dan 1 buah pin sebagai pegalamatan dalam protocol komunikasi I2C. Sensor ini beroperasi menggunakan tegangan sebesar 5V.

Tabel 2.4 Spesifikasi Pin Sensor MPU-6050

Nama Pin	Fungsi
VCC	Sumber arus tegangan untuk mengoperasikan sensor sebesar 5V.
GND	Merupakan <i>ground</i> dari arus tegangan.
SCL	Serial Clock sebagai <i>clock</i> pada komunikasi secara serial.
SDA	Serial Data sebagai pin untuk mengirimkan data secara serial.
XDA	Auxiliary Serial Data
XCL	Auxiliary Serial Clock
ADO	Sebagai pengalaman untuk komunikasi menggunakan protocol I2C.
INT	Sebagai <i>Interrupt</i> pada sensor.

#### A. Accelerometer Pada MPU 6050

Sebuah accelerometer bekerja berdasarkan prinsip efek piezoelektrik. Cara kerjanya seperti kotak cuboidal dengan bola kecil di dalamnya, seperti pada Gambar 2.6



Gambar 2.6 Accelerometer Piezoelektrik

(Sumber <https://www.prometec.net/wp-content/uploads/2015/10/acc.png>)

Dinding kotak tersebut dibuat dengan kristal piezoelektrik. Terjadi perubahan posisi pada kotak, bola terpaksa bergerak ke arah kemiringan, karena gravitasi.

Dinding yang bola bertabrakan dengan menciptakan arus piezoelektrik kecil. Ada tiga pasang dinding yang berlawanan dalam sebuah kubus. Setiap pasangan sesuai dengan sumbu dalam ruang 3D: sumbu X, Y, dan Z. Bergantung pada arus yang dihasilkan dari dinding piezoelektrik, arah kemiringan dan besarnya bisa ditentukan.

### B. Gyroscope Pada MPU 6050

Giroskop bekerja dengan prinsip akselerasi Coriolis. strukturnya mirip garpu yang berada dalam gerakan mundur dan maju konstan seperti pada . Gambar 2.7



Gambar 2.7 Gyroscope Piezoelektrik

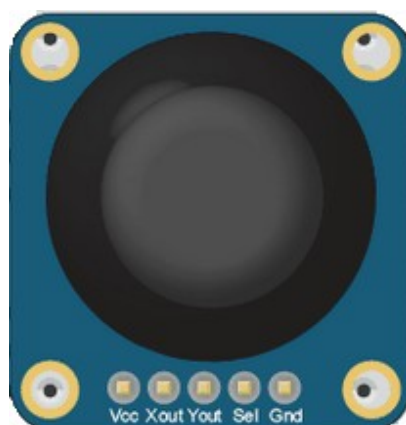
(sumber <https://i.ytimg.com/vi/zwe6LEYF0j8/maxresdefault.jpg>)

Hal ini dilakukan dengan menggunakan kristal piezoelektrik. Kapan pun terjadi kemiringan pengaturan ini, kristal akan mengalami gaya yang berlawanan dengan kecenderungan. Hal ini disebabkan karena inersia garpu bergerak. Kristal menghasilkan arus dalam konsensus dengan efek piezoelektrik, dan arus ini diperkuat. Nilai tersebut kemudian disempurnakan oleh mikrokontroler host.

### 2.2.10 Sensor Joystick

Joystick merupakan perangkat *input* yang memiliki pointer yang dapat digerakkan secara axis 2 dimensi. Joystick menggunakan potensiometer untuk membaca *input* dari pengguna. Potensiometer digunakan untuk mengukur voltase *output* secara *analog* dari sumbu x dan y. Joystick memiliki gagang setir yang dapat bergerak secara axis 2 dimensi x dan y. Gagang setir inilah yang akan mengendalikan perputaran potensiometer pada joystick seperti yang ditampilkan pada Gambar 2.8 yaitu skema sensor joystick tampak atas. Gagang setir pada joystick dapat digerakkan secara diagonal sehingga dapat memberi *input* berupa kombinasi axis x dan y.





**Gambar 2.8 Sensor Joystick**

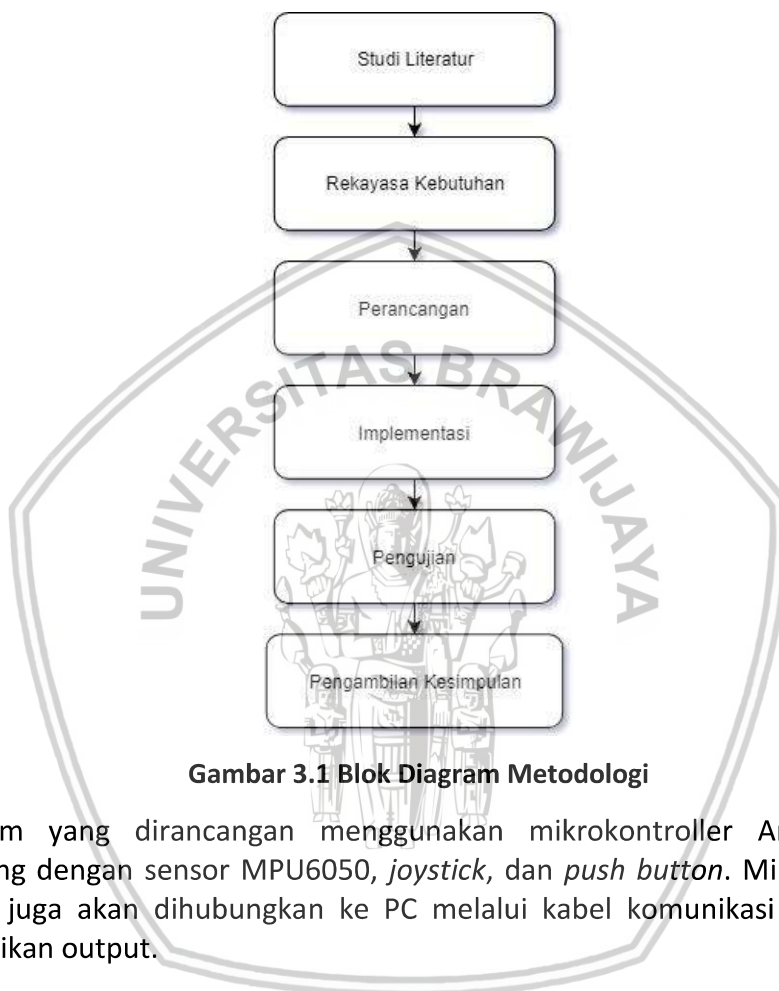
Sensor Joystick memiliki spesifikasi pin seperti yang dijelaskan pada Tabel 2.5 dimana model sensor ini secara umum memiliki 5 pin, yaitu pin Vcc, Gnd, Xout, Yout, dan Sel. Input dari sensor ini ada 3 yaitu Xout dan Yout yang memberikan input berupa axis x dan y serta sel yang berfungsi seperti *button* atau *switch* ketika gagang sensor *joystick* ditekan.

**Tabel 2.5 Spesifikasi Pin Sensor Joystick**

Nama Pin	Fungsi
Vcc	Sebagai sumber arus tegangan untuk mengoperasikan sensor joystick. Voltasenya yaitu 5V.
Gnd	Sebagai <i>ground</i> pada arus tegangan.
Xout	Memberikan output analog dari voltase yang dibaca oleh potensiometer sumbu x. memiliki nilai mulai dari 0 hingga 1024. Nilainya 512 jika dalam keadaan diam.
Yout	Memberikan output analog dari voltase yang dibaca oleh potensiometer sumbu y. memiliki nilai mulai dari 0 hingga 1024. Nilainya 512 jika dalam keadaan diam.
Sel	Merupakan <i>switch</i> yang akan memberikan nilai 1 jika gagang setir ditekan dan 0 saat gagang setir tidak ditekan.

## BAB 3 METODOLOGI

Pada bab ini akan menjelaskan tahapan metode penelitian yang digunakan dan kebutuhan yang diperlukan sistem terkait penelitian yang dilakukan. Penelitian ini bersifat Implementatif. Alur dari tahapan metodologi penelitian yang dilakukan dapat dilihat pada Gambar 3.1



Gambar 3.1 Blok Diagram Metodologi

Sistem yang dirancang menggunakan mikrokontroler Arduino yang terhubung dengan sensor MPU6050, *joystick*, dan *push button*. Mikrokontroler Arduino juga akan dihubungkan ke PC melalui kabel komunikasi serial untuk memberikan output.

### 3.1 Studi Literatur

Studi literatur dilakukan untuk mendapatkan referensi-referensi dari sumber-sumber yang valid atau dari penelitian-penelitian yang telah dilakukan sebelumnya sehingga dapat membantu penulis dalam melakukan penelitian. Dasar-dasar teori yang digunakan oleh penulis diantaranya adalah *Finite State Machine*.

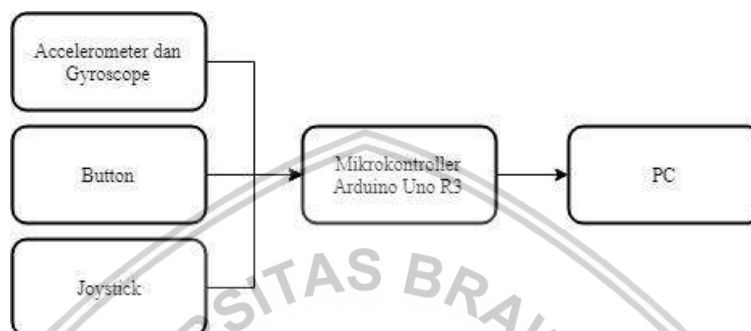
### 3.2 Rekayasa Kebutuhan

Dilakukan untuk menganalisis apa saja yang perlu dipersiapkan dan dibutuhkan untuk penelitian ini. Dapat berupa kebutuhan fungsional maupun non-fungsional.



### 3.3 Perancangan

Sedangkan perancangan sistem dilakukan setelah semua kebutuhan sistem didapatkan melalui tahap analisis kebutuhan yang meliputi kebutuhan perangkat lunak. Setelah kebutuhan sistem terpenuhi selanjutnya adalah perancangan perangkat lunak. Dari hasil rekayasa kebutuhan, didapatkan blok diagram sistem seperti pada Gambar 3.2



Gambar 3.2 Blok Diagram Sistem

Berdasarkan Gambar 3.2 tahapan perancangan sistem yaitu menggunakan Mpu-6050 (*accelerometer* dan *gyroscope*), *Button*, dan *joystick* sebagai input yang datanya akan diproses langsung dengan mikrokontroler Arduino Uno R3. Data tersebut akan dikirimkan ke PC melalui komunikasi serial untuk diolah menjadi input *mouse*.

### 3.4 Implementasi

Pada tahap ini akan dilakukan penerapan dari semua tahapan studi literatur sampai tahap analisis dan perancangan dalam melakukan implementasi terhadap sistem yang dibuat

### 3.5 Pengujian

Pengujian dilakukan untuk mengukur tingkat keberhasilan kebutuhan fungsional dari sistem yaitu mengkonversi vektor 3D dari sensor MPU 6050 ke dalam gerakan 2D Desktop secara akurat menggunakan konversi *pitch* dan *yaw*.

### 3.6 Pengambilan Kesimpulan

Tahap terakhir dari penelitian ini yaitu menarik kesimpulan dari semua tahapan yang telah dilakukan. Selain membuat kesimpulan, saran juga dibuat untuk pengembangan sistem kedepannya.

## BAB 4 REKAYASA KEBUTUHAN

### 4.1 Deskripsi Umum Sistem

Bagian ini menjelaskan secara umum sistem yang akan dibuat oleh penulis. Sistem ini adalah sebuah mouse komputer yang menggunakan accelerometer dan gyroscope sebagai inputnya. Sistem ini dirancang khusus untuk menggantikan mouse pada game bergenre FPS mau game berbasis *First-Person* lainnya.

Sistem yang dibuat akan dihubungkan pada laptop sebagai pengganti mouse. Sistem ini dirancang agar dapat berjalan secara *plug and play* agar tidak memerlukan instalasi tambahan pada laptop yang digunakan untuk menjalankan sistem. Akirlik akan digunakan untuk merancang *case* fisik sistem sehingga akan menyerupai bentuk senjata api yang nyaman di tangan pengguna alat ini.

Berbeda dengan mouse konvensional, sistem ini menggunakan accelerometer dan gyroscope yaitu arah sudut dan orientasi koordinat 3 dimensi axis x, y, dan z. Pergerakan dari sistem yang dibuat akan benar-benar sama dengan pergerakan arah pada game *First-person* yang digunakan. Sistem ini juga memiliki beberapa tombol yang menggantikan fungsi klik kiri dan klik kanan pada mouse. Selain itu tombol tersebut juga ada yang berfungsi sebagai tombol keyboard W, A, S, dan D dimana key tersebut merupakan key untuk bergerak pada game umumnya.

#### 4.1.1 Sampel Game Yang Digunakan

Game yang digunakan sebagai pengujian sistem ini adalah Counter Strike 1.6 yaitu berupa game FPS dan Minecraft yang merupakan game bergenre *first-person*. Tujuan dari menggunakan 2 game berbeda ini adalah untuk mempelajari keselarasan pergerakan mouse pada game first-person berbeda untuk menguji kompatibilitas dari sistem yang dibuat.

Counter Strike 1.6 merupakan game *first person shooter (FPS)* yang tujuan utamanya adalah mengalahkan semua tim lawan ataupun melindungi area yang ditentukan / berhasil memasang bom di area yang ditentukan. Game ini menggunakan mouse untuk kendali cursor senjatanya, klik kiri pada mouse umumnya berfungsi untuk menembak sedangkan klik kanan pada mouse berfungsi untuk membidik. Game ini menggunakan layout tombol keyboard W,A,S,D untuk bergerak dimana W untuk maju, A untuk bergeser ke kiri, D untuk jalan mundur dan S untuk bergeser ke kanan.

Minecraft adalah game *first person* yang beiriontasi *Open World Sandbox*, yaitu pemain bisa melakukan apa saja sebebasnya karena tidak ada tujuan khusus pada game ini. Pada game ini pemain bisa membangun menggunakan balok balok yang tersedia pada game dan juga melakukan sedikit *battle* pada musuh yang ada pada di game. Seperti game *First Person* umumnya, pergerakan game ini menggunakan keyboard W,A,S,D untuk bergerak. Sedangkan klik kiri pada mouse

digunakan untuk berinteraksi pada *item* yang ada pada game, dan klik kanan memiliki fungsi yang bervariasi tergantung *item* yang sedang digunakan pemain.

#### 4.1.2 Ruang Lingkup Sistem

Menjelaskan ruang lingkup tempat berjalannya sistem yang dibuat yang berjudul “Rancang Bangun 3D Oriented Mouse Controller Menggunakan Discrete Closed-Loop Control Berbasis Mikrokontroler Arduino”.

1. Sistem dihubungkan pada port USB pada laptop.
2. Sistem akan dimuat pada sebuah PCB yang menyerupai bentuk senjata api dimana komponen dari sistem ini yaitu sensor MPU-6050, Arduino Uno, Kabel USB type B to USB, dan beberapa button.

#### 4.1.3 Batasan Sistem

Dalam perancangan sistem ini, terdapat batasan-batasan dalam implementasinya. Batasan-batasan tersebut meliputi :

1. Sistem harus mengetahui ukuran resolusi layar yang digunakan pada laptop tempat sistem berjalan.
2. Sensor MPU-6050, button, dan Arduino Uno R3 dimuat ke *case* buatan
3. Digunakan kaca mika untuk membuat *case* yang menyerupai senjata api pada sistem
4. Sistem membaca nilai arah sudut dan point  $x,y,z$  pada sensor MPU 5060
5. Output dari sistem adalah pergerakan arah pada game berbasis *first-person*.

### 4.2 Rekayasa Kebutuhan

Rekayasa Kebutuhan merupakan penjelasan dari semua kebutuhan sistem. Rekayasa Kebutuhan meliputi Kebutuhan Fungsional dan Kebutuhan Non-Fungsional dimana di dalamnya terdapat Kebutuhan Perangkat Keras dan kebutuhan Perangkat Lunak

#### 4.2.1 Kebutuhan Fungsional

Kebutuhan Fungsional adalah kebutuhan-kebutuhan wajib untuk sistem agar dapat beroperasi secara benar, yaitu :

1. Sistem dapat membaca input yang diberikan dengan akurat. Input berupa nilai koordinat dan arah sudut axis  $x,y,z$  dari *accelerometer* dan *gyroscope* pada sensor MPU-5060 yang telah dikonversi menjadi *pitch* dan *yaw*.

2. Sistem menentukan nilai pergerakan dasar pixel dari resolusi layar pada Operating sistem yang digunakan pada laptop terutama ketika mulai memasuki engine dari game berbasis FPS.
3. Tombol tombol pada sistem dapat berfungsi layaknya seperti klik kiri dan kanan pada *mouse*, Tombol W, A, S, D pada *keyboard* komputer.
4. Sistem mampu berkomunikasi dengan komputer yang terhubung melalui komunikasi serial.
5. Semua *state* pada sistem berjalan sesuai kondisi yang telah ditentukan untuk setiap *state*.

#### 4.2.2 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional adalah kebutuhan-kebutuhan yang akan mendukung kinerja sistem. Kebutuhan non-fungsional dibedakan menjadi kebutuhan perangkat keras dan kebutuhan perangkat lunak :

##### 1. Kebutuhan Perangkat Keras

Kebutuhan perangkat keras meliputi laptop, Arduino Uno R3, sensor MPU-6050, dan modul *joystick*. Penjelasan fungsi tiap-tiap komponen dapat dilihat pada Tabel 4.1 yang akan dijelaskan secara detail fungsi dari komponen tersebut.

**Tabel 4.1 Kebutuhan Perangkat Keras Sistem**

Perangkat Keras	Fungsi
Laptop	Digunakan sebagai fasilitas untuk melakukan pemrograman pada mikrokontroller yang digunakan oleh sistem yaitu Arduino Uno. Selain itu laptop juga digunakan sebagai lingkungan pengujian terhadap game yang telah ditentukan..
Arduino Uno	Sebagai mikrokontroller dari sistem untuk mengolah input yang diterima menjadi output yang sesuai.
Sensor MPU-6050	Untuk membaca arah sudut dalam koordinat x,y,z. Merupakan input dari sistem.
Kabel USB type b to USB	Sebagai jalur komunikasi antara laptop yang digunakan dan mikrokontroller.
PCB	Sebagai arus komunikasi pada sistem utama yaitu Mikrokontroller dengan sensor.
Push Button	Beberapa push button yang digunakan pada sistem yang memiliki fungsi berbeda-beda yaitu 2 push button untuk klik kiri dan kanan pada mouse,. 1 push button

	untuk on/off pada sistem, dan 1 push button sebagai penanda kalibrasi pada saat game pertama kali jalan.
Joystick	Sebagai pengendali pergerakan pada game, fungsinya menggantikan tombol W,A,S,D pada keyboard.

## 2. Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak meliputi Operating sistem (Windows 10), Arduino IDE, Counter-Strike 1.6, Minecraft yang fungsinya dapat dilihat pada Tabel 4.2

**Tabel 4.2 Kebutuhan Perangkat Lunak Sistem**

Perangkat Lunak	Fungsi
Operating Sistem	Sebagai pemrosesan dasar pada laptop yang digunakan untuk menjalankan game yang akan diuji.
Arduino IDE	Untuk melakukan pemrograman dan sebagai compiler bahasa c++ yang akan digunakan pada Arduino Uno.
Counter Strike 1.6	Sebagai game berbasis FPS yang akan digunakan dalam pengujian sistem yang dibuat.
Minecraft	Sebagai game berbasis <i>First-Person</i> yang akan digunakan dalam pengujian sistem yang dibuat.
Visual Basic 6	Sebagai IDE untuk membuat software yang akan diinstall pada komputer sebagai jembatan komunikasi antara Arduino dan komputer
Mpu6050.h	Sebuah library untuk mempermudah pembacaan sensor MPU 6050 pada Arduino

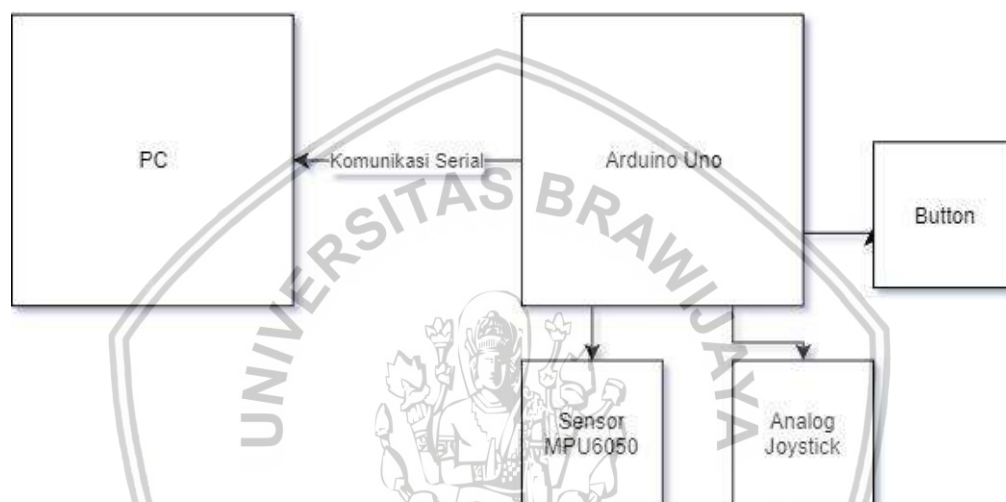


## BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada bab ini dijelaskan deskripsi rancangan dan mengimplementasikan system yang dibuat ke *Personal Computer* menggunakan metodologi penelitian yang telah dibahas.

### 5.1 Perancangan Hardware

Perancangan ini dilakukan untuk menentukan antarmuka fisik dari system yang dikerjakan. Pada poin ini akan dibahas perancangan *case* fisik system serta *hardware* berdasarkan blok diagram pada Gambar 5.1

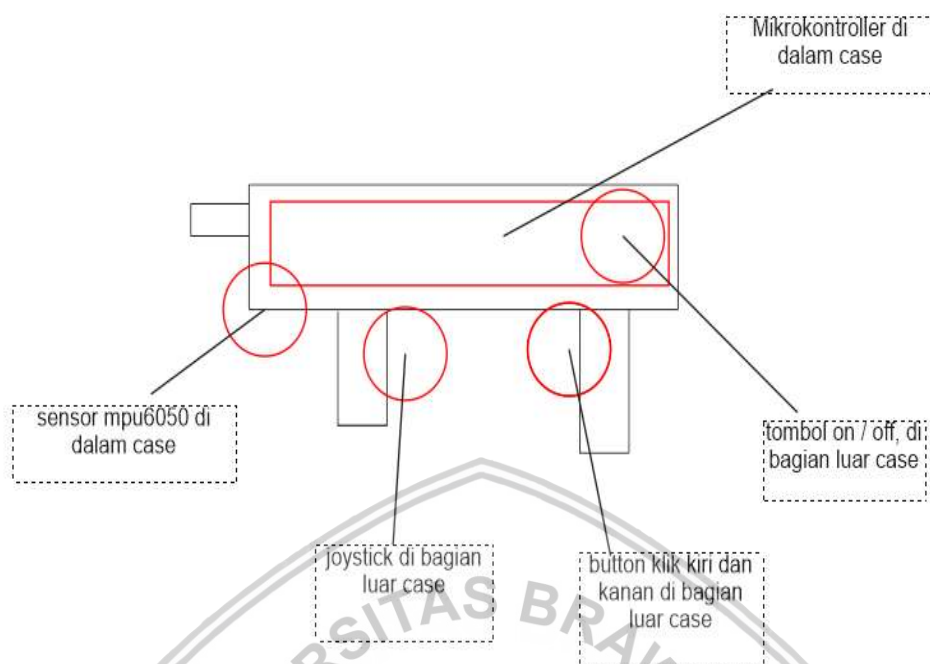


Gambar 5.1 Blok Diagram Sistem

Sistem merupakan Arduino Uno yang dihubungkan dengan sensor MPU6050, button, dan analog joystick. Arduino Uno tersebut nantinya akan dihubungkan pada PC melalui kabel serial dalam mengirimkan datanya pada PC.

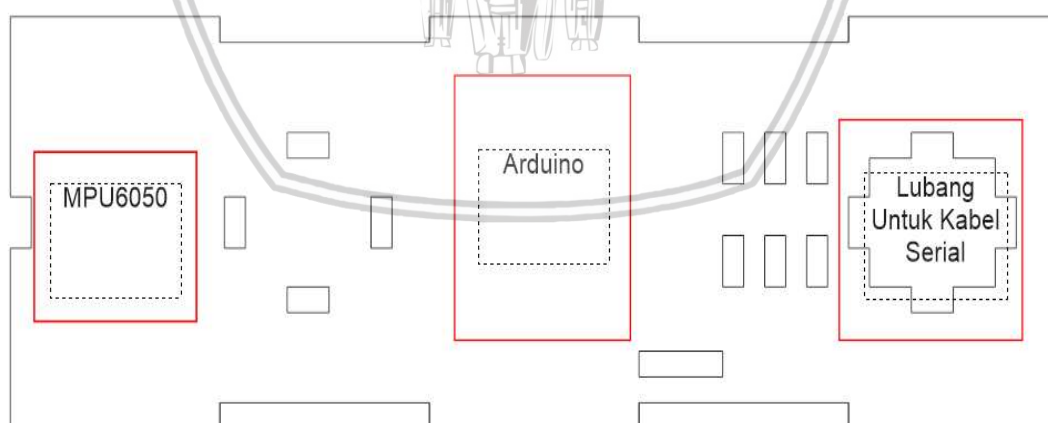
#### 5.1.1 Perancangan Case Fisik Sistem

*Case* fisik dari sistem ini dibuat berdasarkan model senjata api yang memiliki 2 gagang menggunakan mika akrilik setebal 5mm. Akrilik dengan ketebalan 5mm dipilih karena akan lebih mempresentasikan berat dari senjata api sungguhan dan memperkuat *case* alat. Gambaran dasar pada model alat ini ditujukan pada Gambar 5.2. Joystick dan button diletakkan diluar *case* fisik alat karena sensor tersebut memerlukan interaksi langsung dari pengguna. Sedangkan Mikrokontroller Arduino dan sensor MPU-6050 di letakkan di dalam *case*.



**Gambar 5.2 Rancangan Dasar Case Fisik Sistem**

Alat ini menggunakan komunikasi serial, oleh karena itu diperlukan kabel serial untuk menghubungkan Arduino dengan PC. Pada *case* alat perlu dibuat lubang masuk kabel. Gambar 5.3 adalah desain bagian tengah berupa kotak merah pada Gambar 5.2 tampak dari atas. Antara lubang kabel dengan Mikrokontroler Arduino diberi jarak agar saat kabel terpasang, posisinya tidak terlalu menekuk.

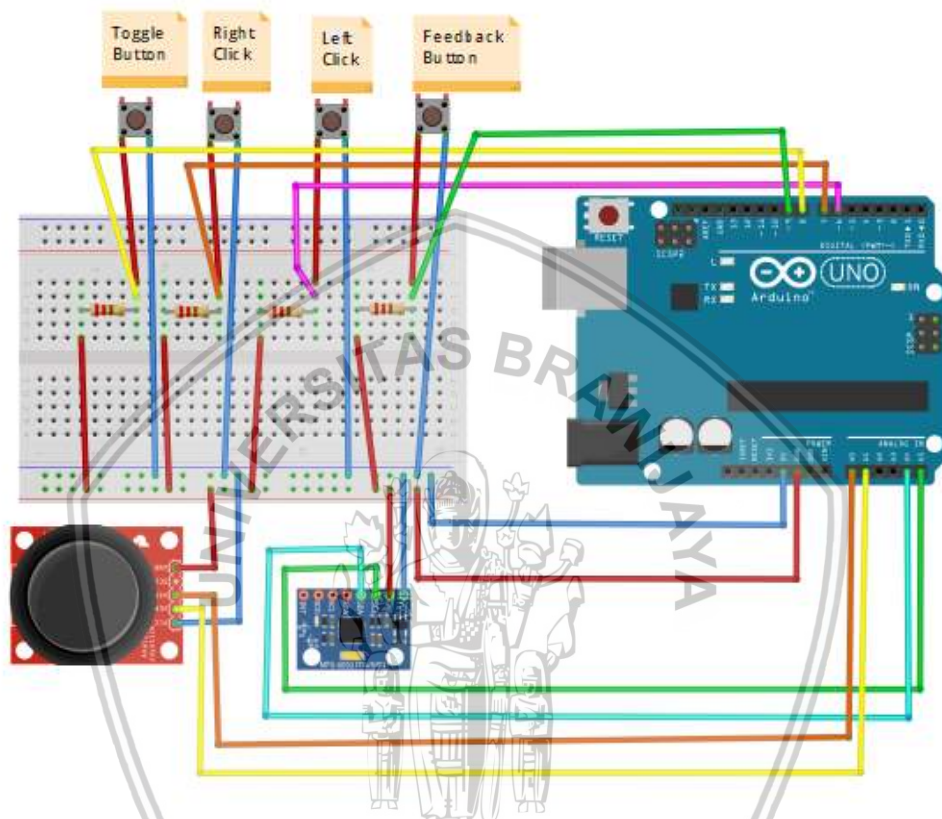


**Gambar 5.3 Tampak Atas Bagian Tengah Case Fisik**

Sensor MPU-6050 diletakkan dibagian depan sedangkan mikrokontroler Arduino terdapat pada bagian tengah. Pada gagang *case* juga disediakan lubang untuk menghubungkan kabel serial dari Arduino ke PC.

### 5.1.2 Perancangan Mikrokontroller Arduino Uno

Mikrokontroller Arduino Uno sebagai pusat pemrosesan data dari sensor dan sebagai input utama dari system yang datanya akan dikirim ke PC. Rancangan dari hardware sistem secara garis besar ditunjukkan pada Gambar 5.4 Dimana mikrokontroller Arduino berperan sebagai pusat *processing* dari system.



Gambar 5.4 Rancangan Hardware Secara Garis Besar

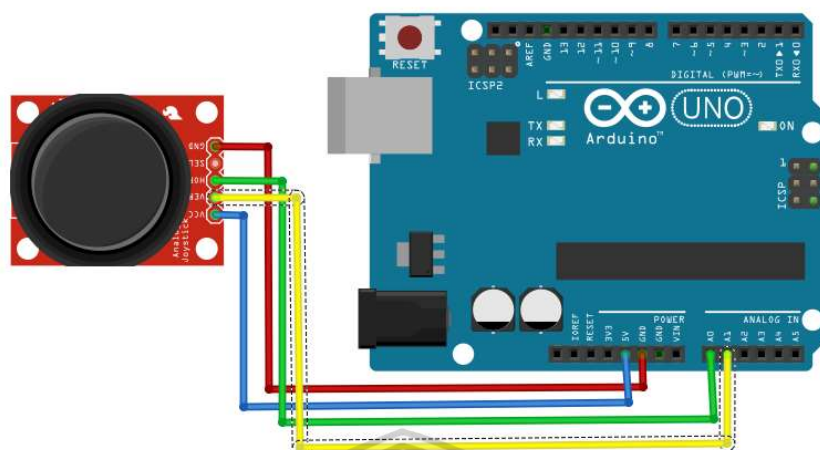
Berdasarkan Gambar 5.4, komponen yang perlu disediakan adalah 1 mikrokontroller Arduino, 4 buah button, 4 buah resistor sebagai rangkaian *pull-down* yang akan dijelaskan lebih detail pada saat membahas rangkaian button, 1 sensor MPU-6050, dan 1 modul analog *joystick*. Spesifikasi dari pin-pin yang digunakan pada Mikrokontroller Arduino dijelaskan pada Tabel 5.1.

Tabel 5.1 Pin Yang Digunakan

Nama Pin	Fungsi
5V	Sebagai sumber tegangan
GND	Sebagai ground

A5	Dihubungkan ke SCL ( <i>Serial Clock</i> ) pada sensor MPU 6050. Merupakan bagian dari protocol komunikasi data I2C dari fungsi <i>wire</i> pada Arduino
A4	Dihubungkan ke SDA ( <i>Serial Data</i> ) pada sensor MPU 6050. Merupakan bagian dari protocol komunikasi data I2C dari fungsi <i>wire</i> pada Arduino
A1	Membaca nilai axis Y yang diberikan oleh sensor joystick
A0	Membaca nilai axis X yang diberikan oleh sensor joystick
Digital 6	Membaca nilai dari button yang berfungsi menggantikan <i>left-click</i> pada Mouse
Digital 7	Membaca nilai dari button yang berfungsi menggantikan <i>right-click</i> pada Mouse
Digital 8	Dihubungkan dengan button yang berfungsi sebagai toggle untuk mengambil alih pergerakan cursor mouse pada PC yang terhubung.
Port Paralel	Digunakan sebagai sumber power pada Arduino Uno R3, selain itu port ini nantinya akan dihubungkan menggunakan kabel "USB to parallel" untuk dihubungkan dengan PC. Selain itu port ini juga berfungsi sebagai jalur komunikasi antara PC dengan Arduino Uno R3.

Sensor joystick memiliki 5 pin yaitu VCC dan GND yang merupakan pin untuk arus tegangan yang digunakan, VrX dan Vry yang merupakan output dari sensor berupa vector x dan y yang direpresentasikan oleh analog pada joystick, dan SW yang merupakan *switch* yang memberikan *logic* ketika analog ditekan. Pada system ini hanya menggunakan pin VCC, GND, Vrx, dan Vry saja yang dihubungkan ke Arduino sesuai dengan Gambar 5.5 dimana Sensor diberi daya tegangan 5 Volt dari mikrokontroller Arduino. Pin SW tidak digunakan karena fungsi untuk melakukan klik kiri dan klik kanan digunakan *button* yang diletakkan pada gagang belakang dari sistem.



**Gambar 5.5 Skema Sensor Joystick dan Mikrokontroler Arduino**

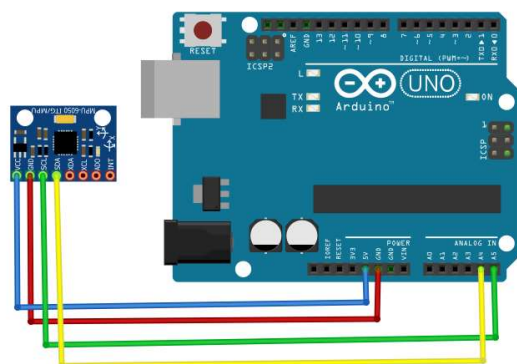
Output dari sensor joystick dihubungkan dengan pin analog pada mikrokontroler yaitu pin Vrx Joystick dengan pin A0 Arduino, Vry dengan pin A1 Arduino sesuai penjelasan pada Tabel 5.2. Hanya saja pada pin SW tidak dihubungkan dengan mikrokontroler Arduino karena fungsionalitas dari pin tersebut tidak diperlukan oleh sistem.

**Tabel 5.2 Pin Sada Sensor Joystick**

Nama Pin	Fungsi
VCC	Merupakan sumber power dari sensor joystick. Dihubungkan dengan tegangan 5V pada Mikrokontroler Arduino.
GND	Merupakan pin <i>ground</i> dari sensor joystick.
Vrx	Merupakan input dari sensor joystick pada axis-x yang memiliki nilai 0 hingga 1024. Nilai saat keadaan diam adalah 512. Dihubungkan ke pin A0 pada mikrokontroler Arduino.
Vry	Merupakan input dari sensor joystick pada axis-y yang memiliki nilai 0 hingga 1024. Nilai saat keadaan diam adalah 512. Dihubungkan ke pin A1 pada mikrokontroler Arduino.
SW	Merupakan switch yang berkerja seperti button, bernilai 1 ketika button ditekan.

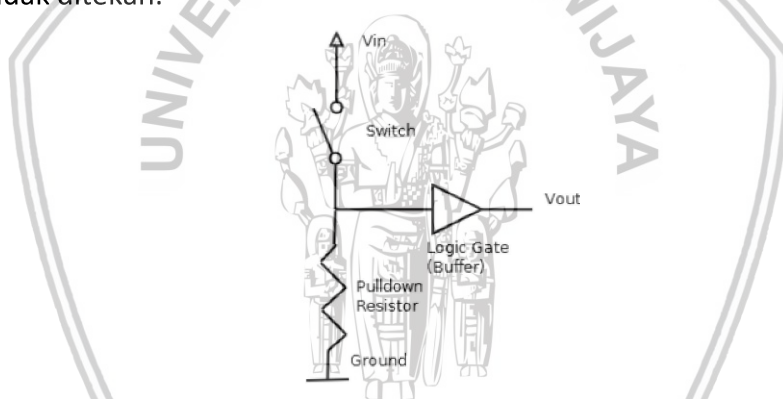
Pada Sensor MPU-6050 pin yang digunakan ada 4 yaitu VCC,GND, SCL, dan SDA. SCL (*Serial Clock Line*) merupakan sinyal *clock* untuk komunikasi I2C dan SDA (*Serial Data Line*) adalah komunikasi datanya. Kedua pin *serial* tersebut harus dihubungkan pada pin analog A4 (SDA), dan A5(SCL) pada Mikrokontroler Arduino Uno seperti rangkaian pada Gambar 5.6 karena pada Mikrokontroler Arduino Uno, pin A4 dan A5 merupakan pin yang digunakan untuk protocol komunikasi I2C.





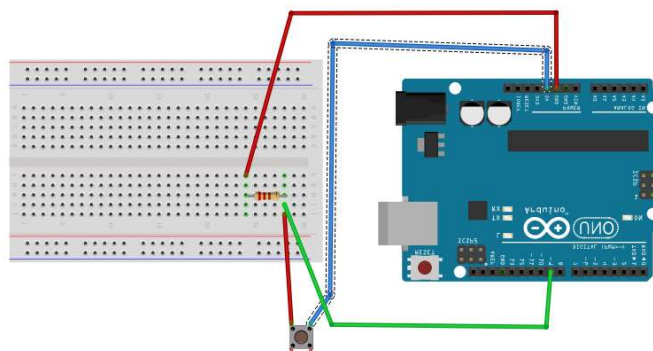
**Gambar 5.6 Koneksi Sensor MPU-6050 Pada Mikrokontroler Arduino**

Sedangkan untuk button dihubungkan dengan pin digital pada Arduino menggunakan *pull-down resistor* seperti pada Gambar 5.7. Hal ini dikarenakan pin digital Arduino memiliki impedansi tinggi sehingga daya berkapasitas kecil dari *noise* lingkungan ataupun sirkuit yang berdekatan dapat mempengaruhi *logic* dari pin digital Arduino. *pull-down resistor* akan membuat *logic* dari button *low* selama button tidak ditekan.



**Gambar 5.7 Rangkaian Pull Down Resistor**

Pada rangkaian Gambar 5.7, resistor diletakkan diantara *logic gate* (button) dan jalur rangkaian yang terhubung dengan *ground*. Button dihubungkan dengan Mikrokontroler Arduino sesuai dengan Gambar 5.8. Pada output yang diberikan dari button selain dihubungkan dengan pin digital pada Arduino, juga dibuat linear dengan resistor 1k ohm yang terhubung dengan arus *ground* sehingga *logic* dari button hanya dapat berubah dengan interferensi langsung saat button di tekan. Pada rangkaian *pull-up* atau *pull-down*, resistor yang digunakan minimal harus 10 kali lebih kecil dari daya impedansi pada rangkaian. Pada rangkaian untuk mengendalikan logika dari *logic gate* / *switch*, resistor yang baik untuk digunakan adalah antara 4,7k ohm hingga 1k ohm.



**Gambar 5.8 Hubungan Button Dengan Mikrokontroler Arduino**

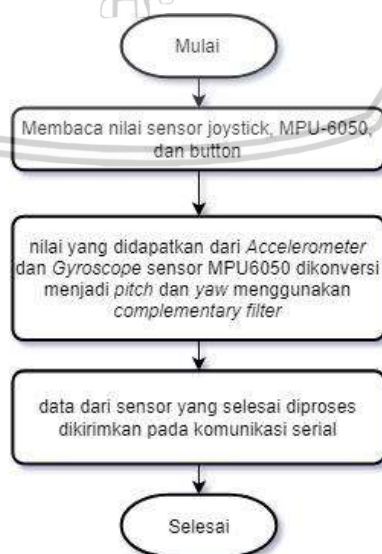
Button hanya memiliki 2 pin, yaitu pin VCC yang merupakan sumber arus dan GND adalah *ground*. Jadi untuk dapat mengambil logika dari button GND dihubungkan pada *pull-down resistor* dan pin digital pada Arduino, sedangkan VCC hanya dihubungkan pada pin VCC 5V Arduino saja.

## 5.2 Perancangan Software

Perancangan ini akan membahas desain dari *flow* software dari sistem yang dibuat. Software menggunakan Bahasa C++ untuk pemrograman Arduino dan Bahasa Visual Basic untuk pemrograman software pada PC.

### 5.2.1 Perancangan Software Pada Sisi Arduino Uno

Fungsi utama dari software untuk memprogram mikrokontroler Arduino adalah untuk melakukan pembacaan nilai sensor, mengolah datanya, dan mengirimkan data yang telah diproses melalui komunikasi serial pada komputer seperti yang diuraikan pada Gambar 5.9.

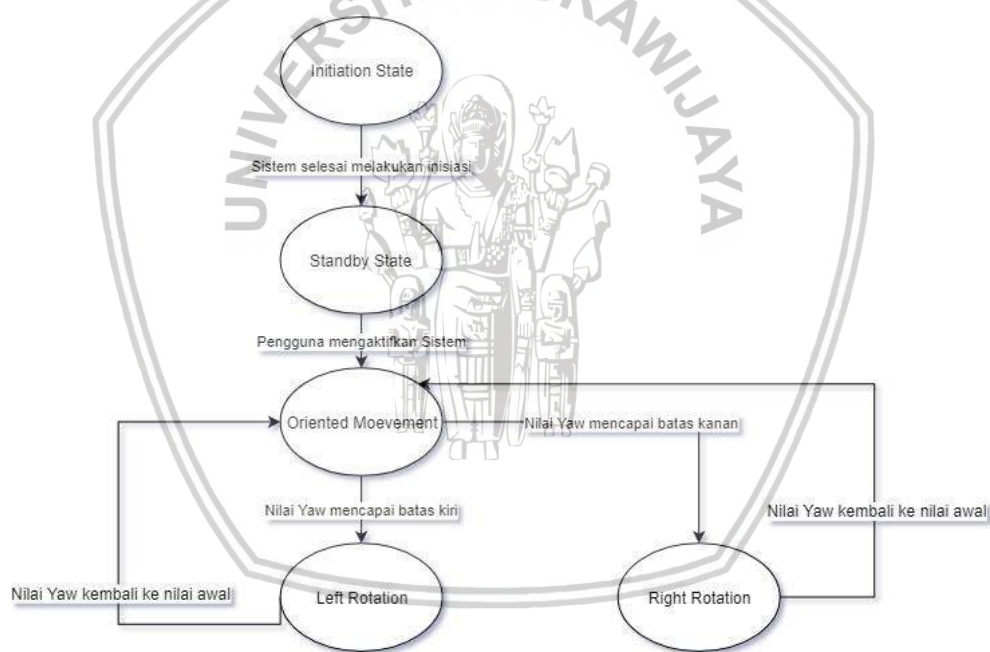


**Gambar 5.9 Flowchart Pada Software Arduino**

Data dari nilai sensor *joystick* yang dibaca berupa nilai axis x dan y pada sensor tersebut. Terdapat 3 buah nilai *button* yang akan dibaca yang masing-masing memiliki fungsi untuk simulasi klik kiri, klik kanan, dan tombol aktivasi dari sistem. Dari sensor MPU-6050 memiliki data berupa orientasi axis x,y, dan z pada *accelerometer* dan *gyroscope* sensor tersebut. Khusus untuk sensor MPU-6050 datanya diproses terlebih dahulu untuk dikonversi menjadi satuan *pitch* dan *yaw*. Data sensor MPU-6050 yang dikirimkan pada komunikasi serial hanya *pitch* dan *yaw* saja.

### 5.2.2 Perancangan Flow Dari State

Pengaplikasian sistem ini menggunakan FSM (*Finite State Machine*) untuk mengatur alur kerja dari sistem. Alur state dapat dilihat pada Gambar 5.10 Flow State Sistem dimana terdapat 3 buah *state* utama yaitu Initiation, Standby, dan Oriented Movement. Pada *state* Oriented Movement terdapat 3 *sub-state* yaitu Center, Left, dan Right.



Gambar 5.10 Flow State Sistem

State dimulai dengan Initiaion State. Initiation State mengatur port, baudrate, dan konfigurasi lainnya yang diperlukan untuk melakukan komunikasi serial. Setelah state ini selesai, berikutnya memasuki Standby State, pada tahap ini sudah terhubung komunikasi antara Mikrokontroler Arduino dan PC melalui kabel serial. Data dari Mikrokontroler Arduino sudah diterima PC namun masih belum di proses. Ketika tombol ON ditekan, barulah data yang diterima diproses dan memasuki state Oriented Movement. Oriented Movement memberikan perpindahan gerak cursor mouse sesuai dengan pergerakan sensor MPU-6050. Ketika arah sudah mencapai titik tertentu di kiri atau kanan maka akan memasuki

Right Rotation / Left Rotation. Pergerakan yang diproses pada state ini merupakan perputaran terus menerus ke kiri atau kanan . semakin jauh arah sudut MPU-6050 dari sudut awal saat tombol ON ditekan, maka perputara semakin cepat.

### 5.2.3 Perancangan Antarmuka

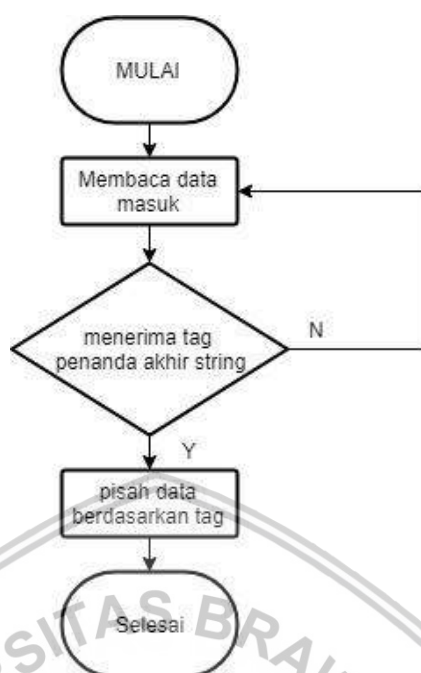
Perancangan ini dilakukan untuk mempermudah membaca nilai yang diterima dari Mikrokontroler Arduino. Data *string* yang diterima akan ditampilkan ke masing-masing kolom data yang bersangkutan. Fungsi dan nilai dari kolom-kolom yang tersedia pada antarmuka software dapat dilihat pada Tabel 5.3

**Tabel 5.3 Kolom Antarmuka Software**

Pitch	Nilai perhitungan pitch dari sensor MPU-6050
Yaw	Nilai perhitungan yaw dari sensor MPU-6050
Joy X	Nilai axis x dari joystick
Joy Y	Nilai axis y dari joystick
Status	Menampilkan nilai 0 dan 1 berdasarkan input dari <i>button</i> yang berfungsi sebagai toggle. Jika nilainya 1, maka fungsi pergerakan cursor mouse pada PC yang terhubung akan diambil alih oleh Mikrokontroler Arduino.
Screen X	Nilai x dari resolusi layer yang dibaca pada PC yang digunakan
Screen Y	Nilai y dari resolusi layer yang dibaca pada PC yang digunakan
Cursor X	Posisi x dari cursor mouse pada layer PC yang digunakan
Cursor Y	Posisi y dari cursor mouse pada layer PC yang digunakan
Left Click	Membaca nilai dari <i>button</i> yang berfungsi menggantikan <i>left click</i> pada mouse. Bernilai 1 ketika <i>button</i> ditekan
Right Click	Membaca nilai dari <i>button</i> yang berfungsi menggantikan <i>right click</i> pada mouse. Bernilai 1 ketika <i>button</i> ditekan
Sensitivity	Nilai bias diatur sesuai kebutuhan pengguna untuk mempengaruhi kecepatan gerak <i>mouse</i> yang disimulasikan oleh sistem.

### 5.2.4 Perancangan Komunikasi Serial

Komunikasi antara Mikrokontroler Arduino dengan PC yang terhubung dilakukan melalui sebuah software yang berfungsi sebagai perantara antara Mikrokontroler dan PC. Software yang digunakan dibuat dengan Visual Basic 6 memiliki *baudrate* maximal 256000. Karena data yang dikirim dari Mikrokontroler Arduino dilakukan secara terus menerus, maka software perlu mengetahui posisi awal dan akhir dari sebuah potongan data dari sebuah string yang diterima terus menerus. Flowchart dari pembacaan data dapat dilihat pada Gambar 5.11

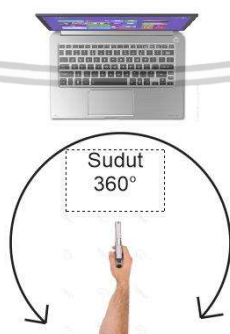


**Gambar 5.11 Flowchart Pembacaan Data Serial**

Selain berfungsi sebagai penanda awal dan akhir suatu sesi data, *tag* juga digunakan untuk memisahkan nilai data yang dikirim oleh setiap sensor yang berbeda pada mikrokontroller Arduino.

### 5.2.5 Perancangan Orientasi Sensor MPU6050

Untuk mempermudah penggunaan data dari sensor MPU-6050, maka nilai mentah dari *Gyroscope* dan *Accelerometer* pada sensor tersebut diubah dalam satuan besaran arah sudut 360 derajat. Ilustrasi orientasi dapat dilihat pada Gambar 5.12. terdapat dua axis sudut yang digunakan, yaitu sudut axis horizontal (*yaw*) sudut axis Vertikal (*pitch*).



**Gambar 5.12 Ilustrasi Orientasi Sensor**

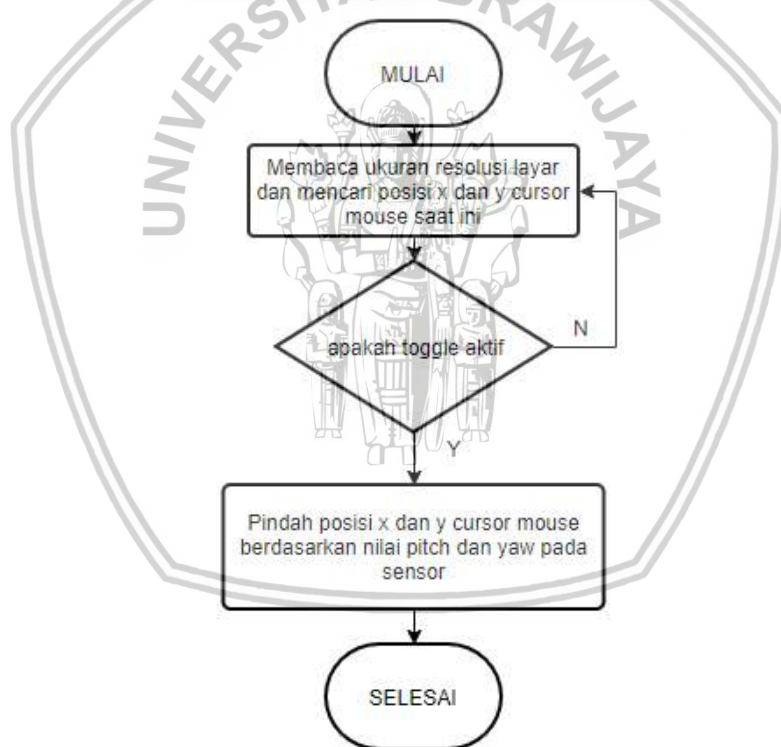
Perpindah sudut inilah yang akan mengatur perpindahan gerak cursor mouse. Tiap x perpindahan sudut dari sensor akan memberikan perpindah sebesar y pada cursor mouse. Nilai variable x, y ini akan diuji untuk menghasilkan penggunaan yang nyaman bagi pemakai alat.



Sensor MPU-6050 memiliki kelemahan yaitu sensor *gyroscope* memiliki respon cepat namun nilainya tidak reliabel dalam penggunaan jangka waktu lama. Sensor *Gyroscope* akan memberikan nilai yang melenceng dalam jangka waktu tertentu yang disebut *drift*. Sedangkan *Accelerometer* nilainya reliabel, namun memiliki respon yang lebih lambat dibanding *Gyroscope*. Mouse konvensional memiliki respon yang sangat cepat dan akurat dalam memberikan nilai perpindahan gerakannya. Oleh karena itu digunakan *Complementary Filter* untuk memberikan nilai yang stabil dengan respon cepat dengan menggabungkan output nilai dari *Accelerometer* dan *Gyroscope* pada MPU-6050.

### 5.2.6 Perancangan Mouse Control

Poin ini merupakan rancangan yang menjelaskan bagaimana Mikrokontroler mengambil alih fungsi *mouse* pada PC yang terhubung dan hubungannya dengan nilai sensor yang terhubung pada Mikrokontroler. Flowchart dari perancangan *mouse* dapat dilihat pada Gambar 5.13



Gambar 5.13 Flowchart Mouse Control

Dalam melakukan pergerakan pixel pada *flowchart* tersebut didapatkan dengan persamaan 5.1 :

$$CX = \frac{RX}{Y_{Max}} \quad CY = \frac{RY}{P_{Max}} \quad (5.1)$$

Pada persamaan 5.1, nilai variabelnya adalah sebagai berikut :

CX = nilai *cursor X*

CY = nilai *cursor* Y  
 RX = ukuran resolusi layar X  
 RY = ukuran resolusi layar Y  
 YMax = nilai maksimal *yaw*  
 PMax = nilai maksimal *pitch*

Sedangkan perpindahan geraknya didapatkan menggunakan persamaan 5.2 :

$$PX = PCX + ((YB - YS) \times CX) \quad PY = PCY + ((PB - PS) \times CY) \quad (5.2)$$

Berdasarkan persamaan 5.2 diketahui variabelnya sebagai berikut :

PX = perpindahan *cursor* X  
 PCX = posisi *cursor* X  
 YB = nilai *yaw* sebelumnya  
 YS = nilai *yaw* saat ini  
 CX = nilai *cursor* X yang didapatkan dari persamaan 5.1  
 PY = perpindahan *cursor* Y  
 PCY = posisi *cursor* Y  
 PB = nilai *pitch* sebelumnya  
 PS = nilai *pitch* saat ini  
 CY = nilai *cursor* Y yang didapatkan dari persamaan 5.1

### 5.2.7 Perancangan W,A,S,D Movement dan Mouse Click

Joystick digunakan sebagai kendali *keyboard* W,A,S,D dan button untuk left dan right click. Sensor joystick memiliki nilai maximum 1024 untuks axis x dan y. nilai ini akan dijadikan pemicu *keyboard* W,A,S,D saat axis x dan y nya berada di pinggir. Sedangkan *button* hanya memiliki nilai diskrit 0 dan 1 untuk *high* and *low*. Pemicu untuk *W,A,S,D movement* dan *mouse click* dapat dilihat pada Tabel 5.4

Tabel 5.4 Pemicu Perintah Keyboard dan Mouse Click

Perintah	Pemicu
Keyboard W	Nilai y dari joystick dibawah 124
Keyboard S	Nilai y dari joystick diatas 900
Keyboard A	Nilai x dari joystick dibawah 124
Keyboard D	Nilai x dari joystick diatas 900
Left-Click	Nilai dari button left-click 1
Right-Click	Nilai dari button Right-click 1

## 5.3 Implementasi Hardware

Pada poin ini akan dibahas pengimplementasian dari Perancangan *Hardware* yang sebelumnya telah dilakukan.

### 5.3.1 Implementasi Case Fisik Dan Mikrokontroller Arduino Uno

Berdasarkan pada rancangan Mikrokontroller Arduino dan rancangan *case* fisik yang dilakukan pada poin 5.1.2 dan 5.1.1 didapatkan hasil implementasi seperti pada Gambar 5.14. Mikrokontroller Arduino dihubungkan dengan Joystick, MPU-6050 dan 4 button pada *case* fisik berbahan akrilik dengan ketebalan 5mm.



(a) Implementasi *Case* Fisik 3D Vector Mouse Controller tampak samping; (b) Implementasi *Case* Fisik 3D Vector Mouse Controller saat sedang dikenakan oleh pengguna.

**Gambar 5.14 mplementasi *Case* Fisik 3D Vector Mouse Controller**

Pada bagian depan gagang belakang diletakkan tombol klik kiri dan kanan untuk menyerupai senjata api sungguhan. Joystick diletakkan pada bagian belakang gagang depan agar mempermudah kendali menggunakan jempol tangan kiri yang memegang gagang depan. Tombol on/off terdapat pada samping *case*. Mikrokontroller Arduino dan sensor MPU-6050 terdapat pada bagian dalam *case*.

## 5.4 Implementasi Software

Pada poin ini akan dijelaskan mengenai implementasi software berdasarkan rancangan software yang telah dilakukan sebelumnya.

### 5.4.1 Implementasi Software Pada Sisi Arduino Uno

*Source code* pemrograman pada *software* mikrokontroller Arduino Uno dalam mengimplementasikan komunikasi serial serta pembacaan sensor dilampirkan pada Tabel 5.5. implementasi dilakukan untuk melakukan pembacaan nilai sensor, mengubah nilai axis dari *accelerometer* dan *gyroscope* menjadi satuan arah *pitch* dan *yaw*. Setelah data pada sisi Arduino selesai diproses selanjutnya data tersebut akan dikirimkan pada software di sisi PC melalui komunikasi serial.

**Tabel 5.5 Source Code Software Mikrokontroller Arduino**

Software pada mikrokontroller Arduino Uno	
1	#include <Wire.h> // memuat modul wire
2	#include <MPU6050.h> // memuat library MPU6050
3	const int mouseButton = 5; // variabel penampung input pin 5
4	const int mouseLeftC = 6; // variabel penampung input pin 6
5	const int mouseRightC = 7; // variabel penampung input pin 7
6	const int modeButton = 8; // variabel penampung input pin 8
7	const int debugButton = 9; // variabel penampung input pin 9
8	const int xAxis = A0; // variabel input axis x joystick pada pin analog A0
9	const int yAxis = A1; // joystick Y axis // variabel input axis x joystick pada pin analog A0
10	MPU6050 mpu; //inisiasi sensor MPU6050
11	int sutep = 0; //deklarasi variabel
12	int Pstatus = 0; //deklarasi variabel
13	int PstatusS = 0; //deklarasi variabel
14	unsigned long timer = 0; //deklarasi variabel
15	float timeStep = 0.01; //deklarasi variabel timestep untuk loop
16	float pitch = 0; //deklarasi variabel yang menampung nilai pitch
17	float yaw = 0; //deklarasi variabel yang menampung nilai yaw
18	float cpitch = 0;
19	float pcoef = 0.98; // variabel coefisien complementary filter
20	int gAccPitch = 0;
21	void setup() {
22	Serial.begin(256000); // deklarasi baud rate komunikasi serial // inisiasi mode range dari sensor MPU-6050 dan pengecekan apakah sensor telah terhubung dengan benar
23	while(!mpu.begin(MPU6050_SCALE_2000DPS, MPU6050_RANGE_2G)){
24	Serial.println("Could not find a valid MPU6050 sensor, check wiring!");
25	delay(500);} //kalibrasi sensor MPU-6050
26	mpu.calibrateGyro();
27	mpu.setThreshold(0.25);}
28	void loop()
29	{timer = millis(); // mengkonversi output dari accelerometer ke dalam pitch

```

30 Vector normAccel = mpu.readNormalizeAccel();
31 float Accpitch = -(atan2(normAccel.XAxis,
    sqrt(normAccel.YAxis*normAccel.YAxis
    + normAccel.ZAxis*normAccel.ZAxis))*180.0)/M_PI;
32 float Accroll = (atan2(normAccel.YAxis,
    normAccel.ZAxis)*180.0)/M_PI;
33 Serial.print("<start>"); // mengirimkan pesan pada komunikasi
    serial
34 int xReading = analogRead(A0); // membaca nilai pin analog A0
35 int yReading = analogRead(A1); // membaca nilai pin analog A1
    // mengirimkan nilai joystick pada komunikasi serial
36 Serial.print("<jyx");Serial.print(xReading);
37 Serial.print("jyx>");
38 Serial.print("<jyy");Serial.print(yReading);
39 Serial.print("jyy>");
    // mengirimkan pesan button yang berperan sebagai klik kiri
40 if (digitalRead(mouseLeftC) == HIGH) {
41 Serial.print("<LF1");Serial.print("LF>");}
42 else if (digitalRead(mouseLeftC) == LOW) {
43 Serial.print("<LF0");Serial.print("LF>"); }
    // mengirimkan pesan button yang berperan sebagai klik kiri
44 if (digitalRead(mouseRightC) == HIGH) {
45 Serial.print("<RG1");Serial.print("RG>"); }
46 else if (digitalRead(mouseRightC) == LOW) {
47 Serial.print("<RG0");Serial.print("RG>"); }
    // mengirimkan nilai dari button yang berfungsi sebagai on/off
    sistem
48 if (digitalRead(modeButton) == HIGH) {
49 if (PstatusS == 0){
50 if (Pstatus == 0){
51 Pstatus = 1;
52 } else {
53 Pstatus = 0;}
54 PstatusS = 1; }
55 Serial.print("<mod");Serial.print(Pstatus);Serial.print("mod>
56 "); } else if (digitalRead(modeButton) == LOW) {
57 PstatusS = 0;
58 Serial.print("<mod");Serial.print(Pstatus);Serial.print("mod>
    "); }

```



```

// inisiasi pembacaan gyroscope dari sensor MPU-6050
59   Vector norm = mpu.readNormalizeGyro();
    //konversi nilai pitch
60   pitch = (1 * (pitch + norm.YAxis * timeStep)) + (0.00 *
    Accpitch);
    //koversi nilai yaw
61   yaw = (1 * (yaw + norm.ZAxis * timeStep));
    //mengirimkan nilai pitch pada komunikasi serial
62   Serial.print("<pitch");
63   Serial.print(pitch);
64   Serial.print("pitch>");
    //mengirimkan nilai yaw pada komunikasi serial
65   Serial.print(" <yaw");
66   Serial.print(yaw);
67   Serial.print("yaw>");
68   Serial.print("<end>\t"); //mengirimkan tag penanda akhir sesi
    pengiriman sebuah paket data
69   Serial.println(" ");
70   delay((timeStep*1000) - (millis() - timer));}

```

Baris 1 dan 2 melakukan *import* pada library yang digunakan pada software Arduino. Pendeklarasian variabel untuk menampung nilai data yang dibutuhkan dilakukan pada baris 3-20. Setelah itu dilakukan deklarasi dari *baudrate* yang digunakan untuk komunikasi serial pada baris 22 yaitu sebesar 256000. Baris 23-27 melakukan kalibrasi dan konfigurasi dari *gyroscope* pada sensor MPU-6050. Baris 28-32 melakukan inisiasi sensor *accelerometer* dari MPU-6050 dan mengkalkulasi nilai *pitch* berdasarkan sensor *accelerometer*. Baris 33 mengirimkan tag penanda dari awal suatu sesi data pada komunikasi serial yaitu "<start>". Pembacaan nilai dari sensor *joystick* berupa nilai axis x dan y dilakukan pada baris 34 dan 35. Pembacaan axis x dilakukan pada pin analog A0 sedangkan nilai axis y pada pin analog A1. Pengiriman data dari nilai sensor joystick kemudian dilakukan pada baris 36-39. Baris 40-47 melakukan pembacaan nilai pada *button* yang berfungsi sebagai klik kiri dan klik kanan serta mengirimkan datanya melalui komunikasi serial. Baris 48-58 melakukan pembacaan terhadap *button* yang berfungsi sebagai tombol aktivasi sistem serta memprogram cara kerja *button* tersebut menjadi seperti *toggle*. Baris 60 melakukan kalkulasi nilai *pitch* dan baris 61 melakukan kalkulasi nilai *yaw*. Baris 62-67 melakukan pengiriman data *pitch* dan *yaw* yang telah selesai diproses. Suatu sesi data berakhir ketiga tag penanda akhir data "<end>" dikirim pada baris 68.

## 5.4.2 Implementasi Flow State Sistem

Dalam implementasi state dari software komunikasi pada sisi PC, flow dari state diletakkan pada penerimaan data yang dibaca oleh komponen *MSCOM* (Kecuali untuk "Initiation" state) karena pada software berbasis *Visual Basic* tidak ada *handle* yang menangani loop seperti pada *Arduino IDE*. State Utama dideklarasikan pada variabel *MajorState* untuk mempermudah membedakan dengan *sub-state*.

Initiation State akan berjalan pada saat software pertama kali dijalankan. State ini berisi rutin untuk mendeklarasikan variabel yang diperlukan, melakukan pembacaan pada port *Serial Com* yang tersedia pada PC dan inisiasi *MSCOM* untuk komunikasi serial. State ini tidak memerlukan mekanisme *looping* sehingga diletakkan pada *Form\_Load* Software seperti yang ditampilkan oleh *source code* pada Tabel 5.6 dimana *Showports* merupakan fungsi buatan yang digunakan untuk menampilkan semua port komunikasi yang tersedia pada PC sistem berjalan, dan deklarasi lainnya merupakan inisiasi variabel *sub-state toggle* yang akan digunakan untuk melakukan pergerakan sesuai input yang diberikan *Arduino Uno*.

Tabel 5.6 Initiation State

Initiation State	
1	Private Sub Form_Load()
2	On Error Resume Next
3	MajorState = "Initiation"
4	Text1.Text = MajorState
5	ShowPorts
6	initCurX = 0
7	initCurY = 0
8	txtScreenX.Text = GetDeviceCaps(Form1.hdc, HORZRES)
9	txtScreenY.Text = GetDeviceCaps(Form1.hdc, VERTRES)
10	MouseToggle = 0
11	WKeyToggle = 0
12	AKeyToggle = 0
13	SKeyToggle = 0
14	DKeyToggle = 0
15	LClickToggle = 0
16	End Sub

Baris 1 adalah awal sub program pada *visual basic* yang menandakan suatu program utama. Baris mendeklarasi variabel *MajorState* yang akan menjadi kondisi yang menampung keadaan state. *Showports* pada baris 5 merupakan fungsi buatan yang digunakan untuk menampilkan semua port komunikasi yang tersedia pada PC sistem berjalan, dan deklarasi lainnya merupakan inisiasi variabel *sub-state toggle* yang akan digunakan untuk melakukan pergerakan sesuai input

yang diberikan Arduino Uno. Pada baris 6 dan 7 di deklarasikan variabel *InitCurX* dan *InitCurY* yang menyimpan posisi *cursor* pada matriks *x* dan *y* di komputer yang digunakan. Baris 8 dan 9 merupakan fungsi Microsoft API untuk mendapatkan ukuran resolusi layar horizontal dan vertical dari komputer yang digunakan. Baris 10-15 mendeklarasikan variabel penampung nilai tombol *keyboard* *W,A,S,D*, klik kiri, dan klik kanan.

Standby State merupakan keadaan dimana PC membaca output dari Arduino Uno namun tidak melakukan pemrosesan data sebelum signature *mod* bernilai 1 (tombol aktif sistem detekan) diterima. Standby State Bersama dengan Active State dimasukkan dalam *handle* MSCOM pada penggalan *source code* dalam Tabel 5.7.

**Tabel 5.7 Standby State**

Standby State	
1	Case comEvReceive ' Received RThreshold # of chars.
2	If (MajorState = "Standby") Or (MajorState = "Active") Then
3	Do
4	InBuff = MSComm1.Input
5	Loop Until MSComm1.InBufferCount < 1
6	If InStr(InBuff, "<start>") And InStr(InBuff, "<start>") Then
7	buffStart = InStr(1, InBuff, "<start>")
8	buffEnd = InStr(1, InBuff, "<end>")
9	If buffEnd > buffStart Then
10	Call HandleInput(InBuff)
11	End If
12	End If
13	End If

Baris 1 merupakan modul *interrupt* dari komunikasi serial COM pada Windows yang akan dijalankan ketika terdapat data yang diterima pada komunikasi serial. Baris 2 percabangan dengan kondisi variabel *MajorState* bernilai "Standby" atau "Active". Baris 4-9 melakukan pembacaan sesi data mulai dari awal tag "<start>" diterima hingga tag "<end>" diterima. Baris 10 memanggil fungsi *HandleInput* yang akan memproses data yang telah diterima.

MSCOM hanya akan mulai melakukan pembacaan data ketika memasuki state "Standby" atau "Active". Active State merupakan keadaan dimana data yang diterima pada komunikasi serial mulai diproses. *Source code* Active State terdapat pada *handle* penerimaan data MSCOM seperti pada Tabel 5.8. Dalam Active State terdapat Sub State "center", "left", dan "right". State "center" akan memproses pergerakan mouse secara orientitas sedangkan "left" dan "right" hanya akan melakukan perputaran secara horizontal secara terus menerus.

Tabel 5.8 Oriented, Left Rotation, Right Rotation State

State Aktif pergerakan mouse	
1	Select Case MajorState
2	Case "Standby"
3	Case "Active"
4	Select Case CurrentState
5	Case "center"
6	MoveMouseCursor xMov, yMov
7	Case "left"
8	RotateFactor = ((LeftRotationStateTrig * 10 / 9) - Fix(CInt(Extraction(fullData, "<yaw", "yaw>")) * 10 / 9))
9	MoveMouseCursor (rect.X + (Abs(RotateFactor) + 1) / 1), yMov
10	Case "right"
11	RotateFactor = (Fix(CInt(Extraction(fullData, "<yaw", "yaw>")) * 10 / 9) - (RightRotationStateTrig * 10 / 9))
12	MoveMouseCursor (rect.X - (Abs(RotateFactor) + 1) / 1), yMov
13	End Select
14	End Select

Baris 1 deklarasi percabangan *case* dengan variabel *MajorState* sebagai pemicu. Baris 2 kondisi jika *case* adalah "Standby". Baris 3 merupakan *case* "Active" dimana di dalamnya terdapa sub-state "center", "left", dan "right". Baris 5-6 isi dari *case* "center" yaitu melakukan pergerakan mouse secara orientasi. *MoveMouseCursor* merupakan fungsi untuk memindahkan *cursor* dari *mouse* ke koordinat x dan y yang telah ditentukan pada komputer. Baris 7-9 merupakan sub-state "left" yang akan melakukan pergerakan mouse ke kiri secara terus menerus. Baris 10-12 adalah sub-state "right" yang melakukan pergerakan ke kanan secara terus menerus.

*Sub-State* dari "center" dimasuki ketika nilai *Pitch* bernilai sama saat seperti sistem memasuki *Active State*. *Rotation angle* pada software merupakan penentu kapan "left" dan "right" state active. *Left State* akan aktif ketikan nilai *Pitch* bernilai kurang dari nilai *pitch* Awal dikurang dengan nilai *textbox* "Rotation Angle" sedangkan *Right State* akan aktif ketikan nilai *Pitch* bernilai lebih dari nilai *pitch* Awal ditambah dengan nilai *textbox* "Rotation Angle" seperti yang terlihat pada source code di Tabel 5.9.

Tabel 5.9 Pemicu Oriented, Left Rotation, Right Rotation State

State Aktif pergerakan mouse	
1	If (MajorState = "Active") And (xMov <= (CInt(txtScreenX.Text))) And (yMov <= (CInt(txtScreenY.Text))) Then
2	If MouseToggle = 0 Then

3	CenterStateTrig = Fix(CInt(Extraction(fullData, "<yaw", "yaw>")))
4	LeftRotationStateTrig = Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) - CInt(txtRotAngle.Text)
5	RightRotationStateTrig = Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) + CInt(txtRotAngle.Text)
6	MouseToggle = 1
7	CurrentState = "center"
8	Else
9	If Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) <= (CenterStateTrig + 6) And Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) >= (CenterStateTrig - 6) Then
10	If CurrentState <> "center" Then
11	CurrentState = "center"
12	End If
13	ElseIf Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) <= LeftRotationStateTrig Then
14	If CurrentState <> "left" Then
15	CurrentState = "left"
16	End If
17	ElseIf Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) >= RightRotationStateTrig Then
18	If CurrentState <> "right" Then
19	CurrentState = "right"
20	End If
21	ElseIf Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) < RightRotationStateTrig And Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) > LeftRotationStateTrig And Fix(CInt(Extraction(fullData, "<yaw", "yaw>"))) <> CenterStateTrig Then
22	End If

Baris 1-2 merupakan percabangan dengan kondisi *State* adalah "Active" dan posisi cursor mouse berada di tengah (posisi awal saat sistem diaktifkan). Baris 3-5 membaca nilai awal *cursor* mouse dan mengkalkulasi kondisi untuk dijadikan pemicu dari *state* Center, Left, dan Right. Baris 6 membaca apakah sistem sudah diaktifkan. Baris 7 mengubah kondisi *state* menjadi Center. Baris 9-12 membaca nilai *yaw* apakah nilainya memiliki perbedaan sebesar 9 dari nilai awal saat sistem dinyalakan. Jika terpenuhi maka *state* akan diubah menjadi Center. Baris 13-23 melakukan pembacaan nilai *yaw* apakah memenuhi kondisi dari *state* Left atau Right. Jika terpenuhi akan mengubah nilai *state* menjadi *state* yang bersangkutan.



### 5.4.3 Implementasi Antarmuka Software

Berdasarkan perancangan antarmuka yang telah dilakukan sebelumnya. Didapatkan antarmuka software antarmuka *software* seperti pada Gambar 5.15. Pitch dan Yaw berisi nilai berupa satuan sudut dari sensor MPU-6050 dengan nilai decimal hingga 2 angka dibelakang koma. Joy X and Joy Y masing-masing merupakan nilai axis dari horizontal dan vertical sensor joystick yang memiliki nilai integer antara 0 hingga 1023 (10 bit). Status mengindikasikan apakah sistem sedang mengambil alih fungsi mouse atau tidak layaknya tombol on/off. Sensitivity merupakan faktor pengali kecepatan gerak perpindahan cursor mouse pada sistem yang dibuat, pengguna bisa memasukkan nilai sensitifitas secara manual.

The screenshot shows a Windows form titled 'Form1'. It contains several input fields and buttons. The fields are: 'Initiation' (text box), 'Pitch' (text box), 'Yaw' (text box), 'Joy X' (text box), 'Joy Y' (text box), 'Status' (text box), 'Screen X' (text box with value 1366), 'Screen Y' (text box with value 768), 'Cursor X' (text box), 'Cursor Y' (text box), 'Left Click' (text box), 'Right Click' (text box), 'Sensitivity' (text box with value 2.5), and 'Rotation Angle' (text box with value 55). There are also buttons for 'Increase', 'Decrease', 'Refresh Port', and 'Select Port'.

Gambar 5.15 Antarmuka Software

### 5.4.4 Implementasi Komunikasi Serial

Untuk melakukan komunikasi antara system yang dibuat dengan PC digunakan komunikasi serial dengan baudrate 256000. Nilai tersebut merupakan baudrate tertinggi yang dapat ditangani Komunikasi Serial berbasis software Visual Basic 6. Baudrate tertinggi dipilih karena system memerlukan respon yang tinggi untuk performanya. Walau pada IDE Arduino tidak tersedia baudrate dengan nilai tersebut, namun komunikasi serial berjalan normal antara Mikrokontroler Arduino dengan PC yang digunakan. Pada software yang dijalankan pada PC menggunakan komponen MSCOMM Microsoft untuk melakukan komunikasi serial dengan konfigurasi *Source Code* yang ditunjukkan pada penggalan *source code* Tabel 5.10.

Tabel 5.10 Konfigurasi Komunikasi Serial

Setting MSCOM	
1	Dim SelectedPort As Integer
2	SelectedPort = Comb01.ListIndex
3	If (SelectedPort < 0) Then
4	Else

5	If InStr(Combo1.Text, "COM") Then
6	SelectedPort = CInt(Mid(Combo1.Text, 4, 1))
7	End If
8	End If
9	MSComm1.RThreshold = 80
10	MSComm1.Settings = "256000,n,8,1"
11	MSComm1.CommPort = SelectedPort
12	MSComm1.PortOpen = True
13	MSComm1.DTREnable = False

Baris 1 mendeklarasikan variabel *integer* SelectedPort. Variabel ini akan menampung nomor dari port pada Windows yang terhubung dengan mikrokontroler Arduino. Baris 2-8 adalah percabangan yang memeriksa apakah *port* yang dipilih pada software sudah valid. Baris 9-13 merupakan konfigurasi untuk komunikasi serial pada Windows. RThreshold adalah ekpektasi Panjang sesi data dalam satuan bit yang akan diterima. Pada Settings 256000 merupakan *baudrate* yang digunakan dalam komunikasi serial. CommPort merupakan nomor dari *port* yang akan digunakan yaitu nomor *port* yang ditampung variabel SelectedPort pada program.

Data yang dikirim melalui Mikrokontroler Arduino merupakan string yang berisi nilai pembacaan dari sensor Mpu 6050, joystick, dan button. String yang dikirim diberi tag untuk mempermudah pembacaan nilainya melalui komunikasi serial. Contoh data string yang dikirim dari Mikrokontroler Arduino dapat dilihat pada Gambar 5.16



**Gambar 5.16 Data String Dari Mikrokontroler Arduino**

Fungsi dari masing-masing tag pada data string yang dikirim oleh Mikrokontroler Arduino dapat dilihat pada Tabel 5.11

**Tabel 5.11 Tag Pada String**

start	Sebagai penanda awal dari string yang dikirim
jyx	Nilai x dari sensor joystick
jyy	Nilai y dari sensor joystick
LF	Nilai dari button yang berfungsi sebagai <i>left click</i>
RG	Nilai dari button yang berfungsi sebagai <i>right click</i>

mod	Nilai dari button mod
pitch	Nilai perhitungan pitch dari sensor MPU 6050
yaw	Nilai perhitungan yaw dari sensor MPU 6050
end	Menandakan akhir dari string

Penanganan data pada software dilakukan dengan pembacaan tag untuk mengetahui kapan sebuah string telah sepenuhnya dikirim dalam komunikasi serial. Fungsi yang mengolah data ketika sebuah tag penanda akhir dari string pada software seperti pada Tabel 5.12.

**Tabel 5.12 Fungsi Pembaca Akhir Tag**

Penggalan source code pada MSCOM event receive.	
1	Case comEvReceive ' Received RThreshold # of chars.
2	If (MajorState = "Standby") Or (MajorState = "Active") Then
3	Do InBuff = MSComm1.Input
4	Loop Until MSComm1.InBufferCount < 1
5	If InStr(InBuff, "<start>") And InStr(InBuff, "<end>") Then
6	buffStart = InStr(1, InBuff, "<start>")
7	buffEnd = InStr(1, InBuff, "<end>")

Pada baris 1 *comEvReceive* merupakan *interrupt* pada komunikasi serial setiap data diterima. Baris 4-7 data akan terus dibaca sampai menemukan tag "<start>" dan "<end>". Ketika tag tersebut diterima, maka data akan di proses untuk memisahkan nilai masing-masing tag.

### 5.4.5 Implementasi Mouse Control

Pergerakan *cursor* x dan y para *mouse* ditangani oleh software. *source code* yang menandai pergerakan mouse dapat dilihat pada Tabel 5.13. potongan *source code* ini melakukan perhitungan matematis untuk mengkonversi nilai *pitch* dan *yaw* kedalam axis x dan y pada komputer. Selain itu juga melakukan pembacaan posisi baru dari axis x dan y pada komputer untuk dibandingkan dengan posisi axis yang sebelumnya. Hal ini dilakukan untuk mempermudah perhitungan pergerakan *cursor* mouse.

**Tabel 5.13 Penanganan Mouse Pada Software**

Penggalan source code pada MSCOM event receive.	
1	If Fix(initCurX) <> Fix(newCurX) Then
2	If initCurX > newCurX Then
	'right
3	xDiff = Fix(CInt(initCurX)) - Fix(CInt(newCurX))

```

4      xMov = rect.X + ((Abs(xDiff)) * Fix(CInt(txtSenv.Text)))
5  Else
6      'left
7      xDiff = Fix(CInt(newCurX)) - Fix(CInt(initCurX))
8      xMov = rect.X - ((Abs(xDiff)) * (CInt(txtSenv.Text)))
9  End If
10     initCurX = newCurX
11 End If
12 If Fix(initCurY) <> Fix(newCurY) Then
13     If initCurY > newCurY Then
14         'up
15         yDiff = Fix(CInt(initCurY)) - Fix(CInt(newCurY))
16         yMov = rect.Y - ((Abs(yDiff)) * (CInt(txtSenv.Text)))
17     Else
18         'down
19         yDiff = Fix(CInt(newCurY)) - Fix(CInt(initCurY))
20         yMov = rect.Y + ((Abs(yDiff)) * (CInt(txtSenv.Text)))
21     End If
22     initCurY = newCurY
23 End If
24 If (MajorState = "Active") And (xMov <= (CInt(txtScreenX.Text)))
25     And (yMov <= (CInt(txtScreenY.Text))) Then
26     If MouseToggle = 0 Then
27         CenterStateTrig = Fix(CInt(Extraction(fullData, "<yaw", "yaw>")))
28         LeftRotationStateTrig = Fix(CInt(Extraction(fullData,
29             "<yaw", "yaw>"))) - CInt(txtRotAngle.Text)
30         RightRotationStateTrig = Fix(CInt(Extraction(fullData, "<yaw",
31             "yaw>"))) + CInt(txtRotAngle.Text)
32         MouseToggle = 1

```

Baris 1 merupakan percabangan dengan kondisi posisi axis x yang baru berbeda dengan posisi axis x yang sebelumnya. Baris 2-4 menggerakkan *cursor mouse* ke arah kanan jika posisi axis x yang baru lebih besar dari axis x yang sebelumnya. Baris 6-8 menggerakkan *cursor mouse* ke arah kiri jika posisi axis x yang baru lebih kecil dari axis x yang sebelumnya. Baris 12-14 menggerakkan *cursor mouse* ke atas jika posisi axis y yang baru lebih kecil dari axis y yang sebelumnya. Baris 16-18 menggerakkan *cursor mouse* ke bawah jika posisi axis y yang baru lebih besar dari axis y yang sebelumnya.

Pergerakan mouse ke kiri dilakukan jika nilai yaw saat ini lebih kecil dari nilai yaw sebelumnya sedangkan pergerakan mouse ke kanan dilakukan jika nilai yaw

saat ini lebih besar dari nilai yaw sebelumnya. Pergerakan mouse ke atas dilakukan jika nilai pitch saat ini lebih kecil dari nilai pitch sebelumnya sedangkan pergerakan mouse ke bawah dilakukan jika nilai pitch saat ini lebih besar dari nilai pitch sebelumnya.

Sensitifitas nilai pergerakan mouse didapatkan dengan cara membagi nilai x dan y pada resolusi layar dengan 360, sehingga didapatkan nilai pixel yang harus dihitung setiap terjadi perpindah gerak yaw dan pitch pada sensor.

#### 5.4.6 Implementasi Orientasi MPU-6050

Pergerakan secara Orientitas hanya dilakukan pada saat kondisi *Sub-State* dari *Active State* adalah *center* Seperti pada Tabel 5.14

**Tabel 5.14 Pergerakan Mouse sesuai State**

Penggalian source code yang mengendalikan inut mouse	
1	Select Case MajorState
2	Case "Standby"
3	Case "Active"
4	Select Case CurrentState
5	Case "center"
6	MoveMouseCursor xMov, yMov
7	Case "left"
8	RotateFactor = ((LeftRotationStateTrig * 10 / 9) -
9	Fix(CInt(Extraction(fullData, "<yaw", "yaw>")) * 10 / 9))
	MoveMouseCursor (rect.X + (Abs(RotateFactor) + 1) / 1), yMov
10	Case "right"
11	RotateFactor = (Fix(CInt(Extraction(fullData, "<yaw", "yaw>")) * 10 / 9) - (RightRotationStateTrig * 10 / 9))
12	MoveMouseCursor (rect.X - (Abs(RotateFactor) + 1) / 1), yMov
13	End Select
14	End Select

Baris 1-4 adalah percabangan *case* pada *flow* pemrograman. Baris 5-6 merupakan keadaan dimana kondisi *sub-state* saat ini Center dan melakukan pergerakan *cursor mouse* secara orientitas. Baris 7-9 melakukan pergerakan kearah kiri secara terus menerus ketika kondisi *sub-state* adalah Left. Baris 10-12 melakukan pergerakan kearah kanan secara terus menerus ketika kondisi *sub-state* adalah Right.

Ketika *sub-state center* pergerakan nilai mouse dilakukan sesuai dengan representasi perubahan nilai *pitch* dan *yaw* dari Arduino Uno. Pada saat *sub-state left* atau *right* nilai *yaw* tidak akan diproses dan pergerakan *pitch* pada Mouse akan



bergerak semakin cepat sesuai dengan seberapa tajam sistem mengarah ke kiri atau kanan.

#### 5.4.7 Implementasi W,A,S,D Movement dan Mouse Click

Perintah yang diterima computer saat keyboard ditekan maupun mouse diklik ada 2. Yaitu *key\_pressed* ketika tombol sedang ditahan dan *key\_released* ketika tombol dilepaskan. Jika perintah *key\_pressed* dikirim tanpa diakhiri dengan - *Key\_released* , akan menghasilkan error pada input computer yaitu *key* tersebut dianggap sedang dalam keadaan ditekan sehingga ketika dikirimkan perintah *keyboard* lain, input yang muncul adalah kombinasi antara *key* yang baru dikirim dengan *key* sebelumnya. Untuk itu, pengiriman perintah *keyboard* dan *mouse click* melalui sebuah software harus bias mensimulasikan *key\_pressed* dan *key\_released* seperti keyboard dan mouse sungguhan.

Pada *key* keyboard dan *mouse click* digunakan *state* untuk mensimulasikan *key\_pressed* dan *key\_released* pada joystick. Pemicu state dapat dilihat pada Tabel 5.15

Tabel 5.15 State Pada Software

W_pressed	Jika nilai axis y joystick dibawah 124
W_released	Jika nilai axis y joystick diatas 124 dan state sebelumnya adalah W_pressed
S_pressed	Jika nilai axis y joystick diatas 900
S_released	Jika nilai axis y joystick dibawah 900 dan state sebelumnya adalah S_pressed
A_pressed	Jika nilai axis x joystick dibawah 124
A_released	Jika nilai axis x joystick diatas 124 dan state sebelumnya adalah W_pressed
D_pressed	Jika nilai axis x joystick diatas 900
D_released	Jika nilai axis x joystick dibawah 900 dan state sebelumnya adalah S_pressed
Left_Click_pressed	Jika button <i>left click</i> bernilai <i>high</i>
Left_Click_released	Jika button <i>left click</i> bernilai <i>low</i> dan state sebelumnya adalah Left_Click_pressed
Right_Click_pressed	Jika button <i>right click</i> bernilai <i>high</i>
Right_Click_released	Jika button <i>right click</i> bernilai <i>low</i> dan state sebelumnya adalah Right_Click_pressed

*Source code* dari software yang menangani state untuk joystick dapat dilihat pada Tabel 5.16 dimana akan membaca *tag* dari joystick untuk mendapatkan nilai axisnya lalu memasukkannya ke logika percabangan sebagai pemrosesan *state*.

**Tabel 5.16 Source Code Keyboard**

Keyboard State	
1	If Cint(Extraction(fullData, "<jyx", "jyx>")) <= 124 Then
2	If WKeyToggle = 0 Then
3	SendKey Asc(UCase("w")), True
4	WKeyToggle = 1
5	End If
6	ElseIf Cint(Extraction(fullData, "<jyx", "jyx>")) >= 900 Then
7	If SKeyToggle = 0 Then
8	SendKey Asc(UCase("s")), True
9	SKeyToggle = 1
10	End If
11	ElseIf Cint(Extraction(fullData, "<jyx", "jyx>")) < 900 And
12	Cint(Extraction(fullData, "<jyx", "jyx>")) > 124 Then
13	If WKeyToggle = 1 Then
14	SendKey Asc(UCase("w")), False
15	WKeyToggle = 0
16	ElseIf SKeyToggle = 1 Then
17	SendKey Asc(UCase("s")), False
18	SKeyToggle = 0
19	End If
20	End If
21	If Cint(Extraction(fullData, "<jyy", "jyy>")) >= 900 Then
22	If AKeyToggle = 0 Then
23	SendKey Asc(UCase("a")), True
24	AKeyToggle = 1
25	End If
26	ElseIf Cint(Extraction(fullData, "<jyy", "jyy>")) <= 124 Then
27	If DKeyToggle = 0 Then
28	SendKey Asc(UCase("d")), True
29	DKeyToggle = 1
30	End If
31	ElseIf Cint(Extraction(fullData, "<jyy", "jyy>")) < 900 And
32	Cint(Extraction(fullData, "<jyy", "jyy>")) > 124 Then
33	If AKeyToggle = 1 Then
34	SendKey Asc(UCase("a")), False
35	AKeyToggle = 0
	ElseIf DKeyToggle = 1 Then
	SendKey Asc(UCase("d")), False

36	DKeyToggle = 0
37	End If
38	End If

Baris 1-5 melakukan pengiriman perintah *keyboard W* ketika nilai axis x dari *joystick* dibawah 124. Baris 6-10 melakukan pengiriman perintah *keyboard S* ketika nilai axis x dari *joystick* diatas 900. Baris 11-19 memeriksa apakah nilai axis *joystick* diatas 124 dan dibawah 900. Hal ini dilakukan untuk mengirimka perintah *released* dari tombol *keyboard W* atau *S*. Baris 20-24 melakukan pengiriman perintah *keyboard A* jika nilai axis y joystick diatas 900. Baris 25-29 melakukan pengiriman perintah *keyboard D* jika nilai axis y joystick dibawah 124. Baris 30-38 melakukan pemeriksaan kondisi *joystick* pada axis y untuk melakukan perintah *released* pada tombol *keyboard A* dan *S*.

Potongan *source code* dari software yang menanganin *state* pada *mouse click* seperti pada Tabel 5.17. sistematis dari cara kerja klik mouse dibuat *toggle* dari sisi software untuk mempermudah melakukan pengiriman perintah klik pada komputer.

**Tabel 5.17 Source Code Mouse Click**

Keyboard State	
1	If LClickToggle = 0 Then
2	LeftDown
3	LClickToggle = 1
4	End If
5	ElseIf txtLclick.Text = "0" Then
6	if LClickToggle = 1 Then
7	LClickToggle = 0
8	LeftUp
9	End If
10	End If
	'mouse click right
11	If txtRclick.Text = "1" Then
12	If RClickToggle = 0 Then
13	RightDown
14	RClickToggle = 1
15	End If
16	ElseIf txtRclick.Text = "0" Then
17	If RClickToggle = 1 Then
18	RClickToggle = 0
19	RightUp

Baris 1-4 memeriksa apakah variabel LClickToggle adalah 0. Angka 0 merupakan kondisi saat tombol klik dalam keadaan tidak ditekan dan 1 adalah nilai saat tombol ditekan. Lalu dipanggil fungsi LeftDown yaitu mengirim perintah pada komputer yang menandakan tombol klik kiri sedang ditekan. Baris 6-10 memeriksa apakah tombol klik kiri sudah dilepas dari keadaan sedang ditekan untuk mengirimkan perintah *release* pada klik kiri yang dilakukan dengan memanggil fungsi LeftUp. Baris 11-15 memeriksa nilai dari variabel RClickToggle apakah nilainya sedang 0 lalu memanggil fungsi RightDown yang akan mengirimkan perintah klik kanan pada komputer. Baris 16-19 memeriksa apakah tombol sudah dilepas dari keadaan telah ditekan dan memanggil fungsi RightUp untuk melakukan *release* pada klik kanan.



## BAB 6 PENGUJIAN

Pada bab 6 pengujian akan membahas mengenai pembahasan dari masing-masing fungsionalitas sistem yang terdiri dari pengujian error pada kestabilan fungsionalitas sensor gyro dan accelerometer (MPU-6050), pengujian fungsionalitas sistem apakah sudah dapat memberikan perintah yang sesuai pada lingkungan sistem berjalan yaitu pada Operating System dan Game, dan hasil kuesioner terhadap kepuasan pengguna sistem. Pengujian stabilitas dan *error* pada arah yang ditunjukkan oleh MPU-6050 dilakukan karena input arah memiliki dampak yang krusial pada pergerakan *cursor* dalam game FPS.

### 6.1 Pengujian Kestabilan Dan Akurasi Sensor MPU-6050

Pengujian ini akan menentukan apakah nilai *pitch* dan *yaw* sudah stabil saat sistem dalam keadaan diam dan juga ketepatan nilai *yaw* dalam menunjukan arah.

#### 6.1.1 Tujuan

Untuk menguji kestabilan sensor MPU-6050 yang menghasilkan nilai orientasi dari *gyro* dan *Accelerometer* yang diubah kedalam *pitch* dan *yaw*.

#### 6.1.2 Prosedur Pengujian

Untuk dapat menguji stabilitas sensor MPU-6050 dibutuhkan prosedur-prosedur pengujian yang dapat dilihat sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu mikrokontroler Arduino Uno, sensor MPU-6050, kabel jumper, PC, dan Kabel Komunikasi Serial.
2. Sambungkan pin SCL dan SDA pada sensor MPU-6050 pada pin analog A5 dan pin analog A4 Arduino Uno serta VCC dan GND pada sensor MPU-6050 pada pin VCC dan pin GND Arduino Uno.
3. Buka *software* arduino IDE lalu ketik *source code* untuk membaca nilai sensor MPU-6050 serta kalkulasi *pitch* dan *yaw* yang di filter menggunakan *Complementary Filter*. lalu *upload* ke dalam mikrokontroler Arduino Uno.
4. Arahkan Sistem ke sudut tertentu lalu nyalakan *serial monitor* pada *software* arduino IDE.
5. Amati dan catat setiap keluaran yang dihasilkan sensor MPU-6050 pada *serial monitor*.
6. Dilakukan *sampling* setiap 5 detik dalam periode 20 detik.
7. Bandingkan hasil dari tiap output.
8. Menentukan presentase *error* dan akurasi sensor dengan menggunakan persamaan 6.1 di bawah :



$$PE = \frac{SB}{PA} \times 100\% \quad (6.1)$$

Berdasarkan persamaan 6.1 terdapat keterangan yang dapat diketahui sebagai berikut :

PE = Presentase *error*

SB = Selisih nilai pembacaan

PA = Pembacaan Nilai awal pada saat sistem pertama kali jalan

Sedangkan untuk melakukan pengujian ketepatan pada nilai arah yang ditunjukkan oleh *yaw* menggunakan prosedur sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu mikrokontroler Arduino Uno, sensor MPU-6050, kabel jumper, PC, dan Kabel Komunikasi Serial.
2. Sambungkan pin SCL dan SDA pada sensor MPU-6050 pada pin analog A5 dan pin analog A4 Arduino Uno serta VCC dan GND pada sensor MPU-6050 pada pin VCC dan pin GND Arduino Uno.
3. Buka *software* arduino IDE lalu ketik *source code* untuk membaca nilai sensor MPU-6050 serta kalkulasi *pitch* dan *yaw* yang di filter menggunakan *Complementary Filter*. lalu *upload* ke dalam mikrokontroler Arduino Uno.
4. Gerakkan Sistem ke sudut kelipatan 90 derajat selama 10 kali.
5. Amati dan catat setiap keluaran yang dihasilkan sensor MPU-6050 pada *serial monitor*.
6. Bandingkan hasil dari tiap output.
7. Menentukan presentase *error* dan akurasi sensor dengan menggunakan persamaan 6.2 di bawah :

$$PE = \frac{SB}{PB} \times 100\% \quad (6.2)$$

Berdasarkan persamaan 6.2 terdapat keterangan yang dapat diketahui sebagai berikut :

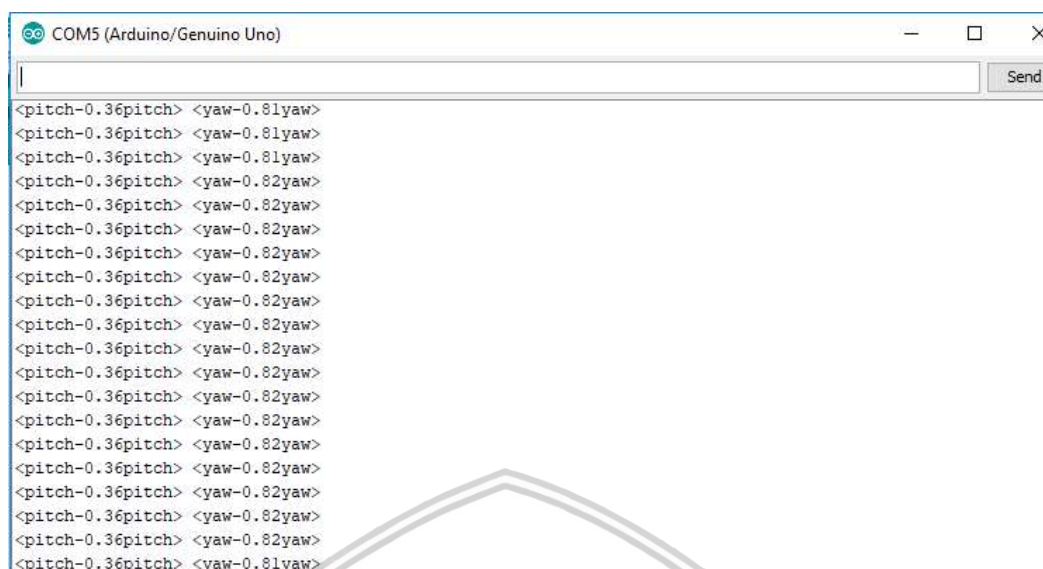
PE = Presentase *error*

SB = Selisih nilai pembacaan

PB = Pembacaan Nilai menggunakan busur

### 6.1.3 Hasil dan Analisis

Pengujian stabilitas sensor MPU-6050 dilakukan selama 2 menit. Berikut merupakan tampilan nilai *pitch* dan *yaw* yang dihasilkan oleh sensor MPU-6050 melalui *serial monitor* setiap *sampling* 5 detik seperti pada Gambar 6.1



```

COM5 (Arduino/Genuino Uno)
Send
<pitch-0.36pitch> <yaw-0.81yaw>
<pitch-0.36pitch> <yaw-0.81yaw>
<pitch-0.36pitch> <yaw-0.81yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.82yaw>
<pitch-0.36pitch> <yaw-0.81yaw>

```

**Gambar 6.1 Hasil Pengujian Stabilitas Nilai**

Pada Gambar 6.1, nilai dibaca oleh sensor MPU-6050 dalam periode 2 menit pada *serial monitor* yang bertujuan untuk menguji kestabilan sensor MPU-6050. Output menghasilkan nilai yang stabil pada *pitch* namun sedikit tidak konsisten pada *yaw* karena sensor *Gyroscope* pada MPU-6050 memiliki drift dan tidak stabil. Nilai *pitch* dan *yaw* pada output tersebut sudah difilter menggunakan *Complementary Filter*.

Berikutnya adalah pengujian ketepatan arah yang ditunjukkan oleh sensor MPU-6050. Walaupun nilai output *yaw* merupakan arah relatif, namun ketepatan ini juga memberikan pengaruh dalam memberikan output dalam berbasis *FPS*. Pengujian dilakukan menggunakan alat ukur busur dimana sistem dalam keadaan berdiri tegak seperti Gambar 6.2.



**Gambar 6.2 Posisi Alat Saat Berdiri Tegak**

Sistem akan diarahkan ke arah kelipatan 90 derajat secara bertahap dan diamati hasilnya. Gambar 6.3 merupakan nilai awal saat sistem dinyalakan.



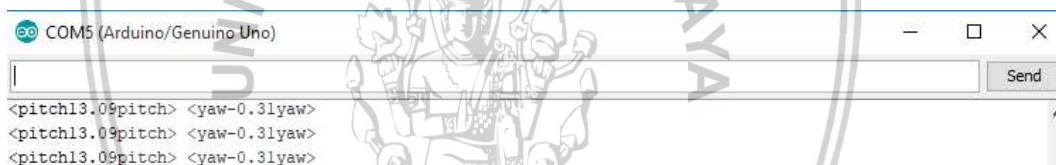
**Gambar 6.3 Nilai Awal Saat Alat Dinyalakan**

Selanjutnya sistem diputar 90 derajat ke arah kiri dan menghasilkan output seperti Gambar 6.4. nilai *yaw* yang diharapkan adalah menjadi 90. Perputaran kekiri menghasilkan nilai plus(+) sedangkan ke kanan menghasilkan nilai minus(-). Namun nilai *yaw* yang dihasilkan sensor MPU-6050 adalah 89,54.



**Gambar 6.4 Alat Diputar 90 Derajat Ke Kiri**

Selanjutnya alat kembali diputar 90 derajat ke arah kanan, maka output *yaw* yang diharapkan adalah kembali menjadi 0 derajat. Hasil output yang ditunjukkan Gambar 6.5 memberikan nilai *yaw* 0,31.



**Gambar 6.5 Alat Diputar 90 Derajat ke Kanan (Sudut Awal)**

Berikutnya alat diputar 90 derajat ke kanan dan diharapkan memberi nilai *yaw* -90. Output alat seperti pada Gambar 6.6 memberikan output meleset yaitu nilai meleset yaitu nilai *yaw* -90,14.



**Gambar 6.6 Alat Diputar 90 Derajat ke Kanan dari sudut 0 derajat**

Terakhir alat diputar lagi 90 derajat ke arah kanan dan outputnya dapat dilihat pada Gambar 6.7 dimana menghasilkan nilai output *yaw* -181,63 sedangkan nilai yang diharapkan adalah -180 derajat



**Gambar 6.7 Alat diputar 90 derajat ke kanan dari sudut -90 derajat**

Percobaan ini diulang selama 10 kali dan diamati hasilnya. Setiap kali pengulangan sistem akan di reset sehingga mengembalikan nilai *yaw* nya menjadi 0. Dari percobaan stabilitas dan keakuratan sensor MPU-6050 didapatkan hasil seperti pada Tabel 6.1 untuk pengujian stabilitas sensor dan table z untuk pengujian keakuratan sensor MPU-6050.

**Tabel 6.1 Hasil Pengujian Stabilitas Sensor MPU-6050**

Detik ke-	Nilai Pitch	Nilai Yaw	Selisih dengan nilai pitch awal	Selisih dengan nilai Yaw awal	Presentase Error (%)
5	0,36	0,81	0	0	0
10	0,36	0,81	0	0	0
15	0,36	0,82	0	0,01	1,23
20	0,36	0,82	0	0,01	1,23
25	0,36	0,82	0	0,01	1,23
30	0,36	0,82	0	0,01	1,23
35	0,36	0,82	0	0,01	1,23
40	0,36	0,82	0	0,01	1,23
45	0,36	0,82	0	0,01	1,23
50	0,36	0,82	0	0,01	1,23
55	0,36	0,82	0	0,01	1,23
60	0,36	0,82	0	0,01	0
65	0,36	0,82	0	0,01	0
70	0,36	0,82	0	0,01	1,23
75	0,36	0,82	0	0,01	1,23
80	0,36	0,82	0	0,01	0
85	0,36	0,82	0	0,01	1,23
90	0,36	0,82	0	0,01	0
95	0,36	0,81	0	0	0
100	0,36	0,81	0	0	0
105	0,36	0,82	0	0,01	1,23
110	0,36	0,82	0	0,01	1,23
115	0,36	0,81	0	0	0
120	0,36	0,81	0	0	0
Rata-Rata	46,5	0,8165	0	0,007	0,861 %

Pada Tabel 6.1 dapat diamati bahwa kestabilan sensor MPU6050 memiliki presentase *error* sebesar 0,861% pada nilai *yaw* sensor tersebut. Hasil presentase tersebut berdasarkan kalkulasi dari persamaan 6.2. yang dijelaskan sebelumnya. Berikutnya untuk hasil uji keakuratan arah *yaw* yang dintunjukkan oleh sensor dapat dilihat pada Tabel 6.2

Tabel 6.2 Hasil Pengujian Akurasi Arah Sudut Yaw

Pengujian ke-	Nilai Yaw Sensor	Nilai yang diharapkan berdasarkan pengukuran alat busur	Persentase Error
1	87,54	90	2,73 %
	-0,31	0	0,31 %
	-90,14	-90	0,16 %
	-182,63	-180	1,46 %
2	91,52	90	1,69 %
	2,05	0	2,05 %
	-89,94	-90	0,07 %
	-178,36	-180	0,91 %
3	90,56	90	0,62 %
	-0,85	0	0,85 %
	-89,77	-90	0,26 %
	-183,66	-180	2,03 %
4	91,33	90	1,48 %
	1,31	0	1,31 %
	-93,71	-90	4,12 %
	-184,97	-180	2,76 %
5	93,71	90	4,12 %
	-2,81	0	2,81 %
	-93,05	-90	3,39 %
	-179,85	-180	0,08 %
6	90,13	90	0,14 %
	0,00	0	0,00 %
	-91,75	-90	1,94 %
	-181,4	-180	0,78 %
7	89,80	90	0,22 %
	-3,10	0	3,10 %
	-87,09	-90	3,23 %



	-175,88	-180	2,29 %
8	90,99	90	1,10 %
	2,95	0	2,95 %
	-92,12	-90	2,36 %
	-178,44	-180	0,87 %
9	93,41	90	3,79 %
	-0,39	0	0,39 %
	-87,95	-90	2,28 %
	-181,00	-180	0,56 %
10	87,95	90	2,28 %
	-1,20	0	1,20 %
	-89,55	-90	0,50 %
	-177,38	-180	1,46% %
Rata-rata	45,10	45	1,61%

Nilai persentase error dari Tabel 6.2 didapatkan menggunakan persamaan 6.2 yang telah dijelaskan sebelumnya. Percobaan ini dilakukan dengan mengarahkan alat ke arah yang telah ditentukan secara bertahap dan diulangi selama 10 kali. Akurasi dari nilai *yaw* sistem memiliki rata-rata error sebesar 1,61%. Dari pengamatan penulis, beberapa faktor yang dapat menyebabkan error adalah bagaimana sensitifitas dari sensor MPU-6050 di program. Sensitifitas tinggi akan menimbulkan nilai yang tidak stabil namun memberi respon cepat, sedangkan sensitifitas rendah memberikan nilai yang stabil namun tidak merespon jika sistem digerakkan dengan sangat pelan.

## 6.2 Pengujian Fungsionalitas *Keyboard Event*, *Mouse Event* dan *Engine Dari Game*

Pengujian dilakukan untuk mengamati perintah yang di kirim oleh alat kepada *Operating System* untuk dibandingkan dengan perintah yang di kirim melalui *keyboard* dan *mouse* sungguhan.

### 6.2.1 Tujuan

Menguji apakah alat mampu memberikan perintah layaknya *keyboard* atau *Mouse* sungguhan terhadap game dan *Operating System*.

### 6.2.2 Prosedur Pengujian

Untuk dapat menguji Fungsionalitas alat harus melakukan tahapan prosedur berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu mikrokontroler Arduino Uno, sensor MPU-6050, kabel jumper, PC, dan Kabel Komunikasi Serial.
2. Sambungkan pin SCL dan SDA pada sensor MPU-6050 pada pin analog A5 dan pin analog A4 Arduino Uno serta VCC dan GND pada sensor MPU-6050 pada pin VCC dan pin GND Arduino Uno.
3. Buka *software* arduino IDE lalu ketik *source code* untuk membaca nilai sensor MPU-6050 serta kalkulasi *pitch* dan *yaw* yang di filter menggunakan *Complementary Filter*. lalu *upload* ke dalam mikrokontroler Arduino Uno.
4. Jalankan Software komunikasi antara Mikrokontroller Arduino dengan Operating Sistem Windows.
5. Bandingkan *message* yang diterima *Windows* melalui *keyboard* dan *mouse* sungguhan dengan alat. Dalam monitoring *event* tersebut menggunakan bantuan dari web:  
*Keyboard* : <https://w3c.github.io/uievents/tools/key-event-viewer.html>  
*Mouse* : <https://w3c.github.io/uievents/tools/mouse-event-viewer.html>
6. Uji Fungsionalitas alat pada lingkungan *operating system* dan saat game Minecraft dan Counter-Strike 1.6 berjalan selama sepuluh kali.

### 6.2.3 Hasil Dan Analisis

Dari perbaan analisis perintah yang dikirim *keyboard* sungguhan didapatkan hasil seperti pada Gambar 6.8 dengan keterangan seperti pada Tabel 6.3.

**Tabel 6.3 Penjelasan Message Event Keyboard**

keydown	Dikirim ke Operating System ketika tombol pada <i>Keyboard</i> ditekan
keydown	Dikirim ke Operating System ketika tombol pada <i>Keyboard</i> telah ditekan dan sedang ditahan
keyup	Dikirim ke Operating System ketika tombol pada <i>Keyboard</i> dilepas
charCode	Kode ASCII dari tiap karakter di <i>keyboard</i>
keyCode	Kode pada <i>event</i> Operating System Windows untuk karakter keyboard (mengabaikan <i>uppercase</i> dan <i>lowercase</i> )
UI Events : key	Hasil output setelah <i>event</i> di proses oleh <i>driver</i> bawaan Operating System berupa karakter.
UI Events : code	Hasil output setelah <i>event</i> di proses oleh <i>driver</i> bawaan Operating System berupa <i>code</i> .

#	Event type	Legacy			UI Events						
		charCode	keyCode	which	key	code	location	repeat	isComposing	inputType	data
12	keyup ↑	0	68 <b>D</b>	68	<b>d</b>	KeyD	0	X	X	-	-
11	keypress	100 d	100	100	d	KeyD	0	X	X	-	-
10	keydown ↓	0	68 <b>D</b>	68	<b>d</b>	KeyD	0	X	X	-	-
9	keyup ↑	0	83 <b>S</b>	83	<b>s</b>	KeyS	0	X	X	-	-
8	keypress	115 s	115	115	s	KeyS	0	X	X	-	-
7	keydown ↓	0	83 <b>S</b>	83	<b>s</b>	KeyS	0	X	X	-	-
6	keyup ↑	0	65 <b>A</b>	65	<b>a</b>	KeyA	0	X	X	-	-
5	keypress	97 a	97	97	a	KeyA	0	X	X	-	-
4	keydown ↓	0	65 <b>A</b>	65	<b>a</b>	KeyA	0	X	X	-	-
3	keyup ↑	0	87 <b>W</b>	87	<b>w</b>	KeyW	0	X	X	-	-
2	keypress	119 w	119	119	w	KeyW	0	X	X	-	-
1	keydown ↓	0	87 <b>W</b>	87	<b>w</b>	KeyW	0	X	X	-	-

Gambar 6.8 Message Event Keyboard Sungguhan

Selanjutnya adalah melakukan monitoring perintah yang dikirim alat untuk dibandingkan dengan perintah yang dikirim melalui *keyboard* sungguhan. Hasil dari *keyboard event* alat dapat dilihat pada Gambar 6.9

#	Event type	Legacy			UI Events						
		charCode	keyCode	which	key	code	location	repeat	isComposing	inputType	data
12	keyup ↑	0	68 <b>D</b>	68	<b>d</b>		0	X	X	-	-
11	keypress	100 d	100	100	d		0	X	X	-	-
10	keydown ↓	0	68 <b>D</b>	68	<b>d</b>		0	X	X	-	-
9	keyup ↑	0	65 <b>A</b>	65	<b>a</b>		0	X	X	-	-
8	keypress	97 a	97	97	a		0	X	X	-	-
7	keydown ↓	0	65 <b>A</b>	65	<b>a</b>		0	X	X	-	-
6	keyup ↑	0	83 <b>S</b>	83	<b>s</b>		0	X	X	-	-
5	keypress	115 s	115	115	s		0	X	X	-	-
4	keydown ↓	0	83 <b>S</b>	83	<b>s</b>		0	X	X	-	-
3	keyup ↑	0	87 <b>W</b>	87	<b>w</b>		0	X	X	-	-
2	keypress	119 w	119	119	w		0	X	X	-	-
1	keydown ↓	0	87 <b>W</b>	87	<b>w</b>		0	X	X	-	-

Gambar 6.9 Message Event Alat

Dari hasil perbandingan pada Gambar 6.8 dan Gambar 6.9 dapat dilihat bahwa semua perintah menghasilkan *event* yang sama kecuali pada bagian UI Event : code. *Event* hanya mampu dihasilkan oleh *keyboard* sungguhan yang memberikan perintah pada *low-level* atau Virtual Keyboard yang memiliki *driver* untuk memberikan perintah pada *low-level*. Dimana pada sistem *mouse-controller* milik penulis tidak menggunakan driver. Berikutnya adalah hasil *monitoring* terhadap *event mouse* sungguhan yang hasilnya dapat dilihat pada Gambar 6.10.



#	Event type	Count	Event							UIEvent	
			eventPhase	bubbles	cancelable	defaultPrevented	composed	isTrusted	timeStamp	view	detail
21	mousemove	2	AtTarget	✓	✓	✗	✓	✓	363339.89999999903	"	0
19	mouseup		AtTarget	✓	✓	✗	✓	✓	362947.79999999993	"	1
18	mousedown		AtTarget	✓	✓	✗	✓	✓	362852.19999999994	"	1
17	mouseup		AtTarget	✓	✓	✗	✓	✓	362259.59999999994	"	1
16	mousedown		AtTarget	✓	✓	✗	✓	✓	362155.89999999973	"	1
15	mousemove	15	AtTarget	✓	✓	✗	✓	✓	361115.49999999999	"	0

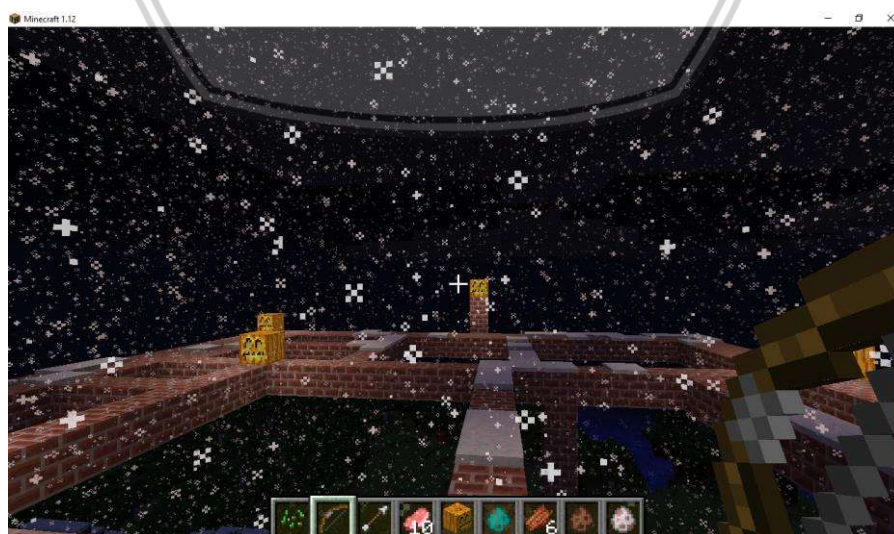
Gambar 6.10 Event Yang Dihasilkan Mouse Sungguhan

Mousemove merupakan *event* yang ditangani oleh Operating System ketika mouse digerakkan sedangkan mousedown dan mouseup adalah ketika tombol *mouse* di klik dan dilepas. *Event mouse* secara *virtual* yang dihasilkan oleh alat dapat dilihat pada Gambar 6.11

#	Event type	Count	Event							UIEvent	
			eventPhase	bubbles	cancelable	defaultPrevented	composed	isTrusted	timeStamp	view	detail
128	mousemove	2	AtTarget	✓	✓	✗	✓	✓	2902781.7999999999	"	0
126	mousemove	1	AtTarget	✓	✓	✗	✓	✓	2902749.8	"	0
125	mousemove	47	AtTarget	✓	✓	✗	✓	✓	2902733.8	"	0
78	mouseup		AtTarget	✓	✓	✗	✓	✓	2895151.19999999983	"	1
77	mousemove	10	AtTarget	✓	✓	✗	✓	✓	2895148.7999999999	"	0
67	mousedown		AtTarget	✓	✓	✗	✓	✓	2894970.9999999995	"	1
66	mousemove	16	AtTarget	✓	✓	✗	✓	✓	2894915.6999999997	"	0
50	mouseup		AtTarget	✓	✓	✗	✓	✓	2892402.0999999999	"	1
49	mousemove	8	AtTarget	✓	✓	✗	✓	✓	2892368.6999999993	"	0
41	mousedown		AtTarget	✓	✓	✗	✓	✓	2892202.1999999997	"	1
40	mousemove	35	AtTarget	✓	✓	✗	✓	✓	2892168.8999999999	"	0

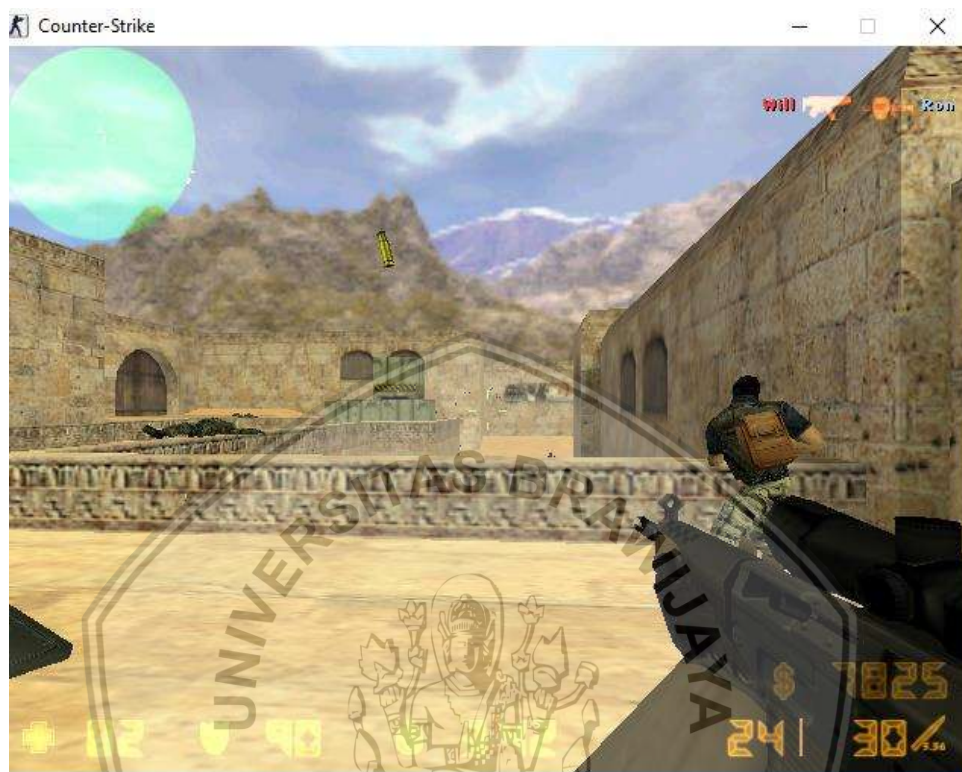
Gambar 6.11 Event Mouse Dari Alat

Dari perbandingan Gambar 6.10 dan Gambar 6.11. *Event* yang dihasilkan oleh mouse sungguhan dan alat *mouse-controller* sama. Hanya saja memiliki *timestamp* yang berbeda. Selanjutnya alat diuji langsung pada lingkungan Operating System, *Engine Game* Minecraft (berbasis OpenGL) pada Gambar 6.12 dan Counter-Strike 1.6 (berbasis OpenGL dan DirectX) pada Gambar 6.13.



Gambar 6.12 Pengujian Alat Pada Game Minecraft

Saat diuji pada *game* Minecraft, semua fungsionalitas alat berfungsi yaitu klik kiri, klik kanan, gerakan *mouse*, dan tombol *keyboard* w,a,s,d. sensitifitas mouse jauh lebih tinggi ketika memasuki *game* Minecraft.



**Gambar 6.13 Pengujian Alat Pada Game Counter Strike 1.6**

Sedangkan pada *game* Counter-Strike 1.6 fungsionalitas tombol *keyboard* alat hanya berfungsi pada main menu karena proteksi dari *engine game* tersebut untuk menghindari *automation*. Proteksi ini dapat di *bypass* jika *event keyboard* yang dihasilkan berlangsung pada *low-level* seperti *keyboard* sungguhan. Dari dari pengujian yang dilakukan sebanyak 10 kali didapatkan hasil seperti pada Tabel 6.4

**Tabel 6.4 Hasil Pengujian Fungsionalitas Alat**

Fungsionalitas Alat	Operating System (Windows 10, Windows 8)	Minecraft (openGL)	Counter Strike 1.6 (DirectX, openGL)
Gerakan Mouse	Berfungsi	Berfungsi	Berfungsi
Klik Kiri	Berfungsi	Berfungsi	Berfungsi
Klik Kanan	Berfungsi	Berfungsi	Berfungsi
Tombol W	Berfungsi	Berfungsi	Tidak Berfungsi
Tombol A	Berfungsi	Berfungsi	Tidak Berfungsi
Tombol S	Berfungsi	Berfungsi	Tidak Berfungsi



Tombol D	Berfungsi	Berfungsi	Tidak Berfungsi
----------	-----------	-----------	-----------------

Pada lingkungan Operating Sistem, untuk bagian *keyboard event* alat akan menampilkan karakter w,a,s,d dalam *lowercase* walaupun kondisi *Caps Lock* pada keyboard sungguhan dalam kondisi aktif. Pada *game* Minecraft semua fungsionalitas berfungsi. Namun pada Counter Strike 1.6 *Keyboard event* tidak berfungsi. Hal ini disebabkan karena beberapa engine game hanya menerima input pada *low-level* dan juga beberapa game memiliki algoritma untuk mencegah *automation* yaitu membedakan *virtual keyboard* dan *keyboard* sungguhan melalui *keycode* yang diterima oleh *operating system* dengan tujuan untuk mencegah *cheater*.

### 6.3 Pengujian User Experience (UX) Terhadap Pengguna Sistem

Pengujian ini dilakukan untuk mengetahui apakah alat sudah layak untuk digunakan sebagai pengganti *mouse* konvensional dalam bermain game FPS. Pengujian ini dilakukan menggunakan standar dari *User Experience (UX)* untuk mengukur nilai kepuasan pengguna.

#### 6.3.1 Tujuan

Mengumpulkan data kepuasan pengguna terhadap alat apakah sudah sesuai dengan kriteria pemain game dan kemudahan dalam penggunaan alat.

#### 6.3.2 Prosedur Pengujian

Pengujian ini dilakukan melalui prosedur berikut :

1. Menyiapkan alat-alat yang dibutuhkan, yaitu mikrokontroler Arduino Uno, sensor MPU-6050, kabel jumper, PC, dan Kabel Komunikasi Serial, PC.
2. Sambungkan pin SCL dan SDA pada sensor MPU-6050 pada pin analog A5 dan pin analog A4 Arduino Uno serta VCC dan GND pada sensor MPU-6050 pada pin VCC dan pin GND Arduino Uno.
3. Hubungkan port Serial pada Arduino Uno dengan PC yang digunakan lalu jalankan *software* komunikasi antara Mikrokontroller Arduino dengan Operating Sistem Windows.
4. Meminta beberapa individe berbeda (30 orang) untuk mencoba alat dengan memainkan game Minecraft.
5. Setelah selesai, masing-masing individu diminta untuk mengisi kuesioner untuk mengumpulkan data yang relevan dengan alat yang dibuat.
6. Menghitung nilai kuesioner untuk setiap beberapa faktor *user experience* yang telah dijelaskan sebelumnya (*Useful*, *Accessible*, *Usable* dan *Desirable*) dengan kriteria :

Sangat Setuju = 10 poin

Setuju = 7,5 poin

Ragu = 5 poin

Tidak Setuju = 2,5 poin

Sangat Tidak Setuju = 0 poin

Lalu digunakan persamaan 6.3 untuk menghitung nilai dari setiap data kuesioner :

$$ND = \frac{TN}{10 \times JR} \times 100\% \quad 6.3$$

ND : Nilai Data

TN : Total Nilai Dalam Suatu Data Kuesioner

JR : Jumlah Responden

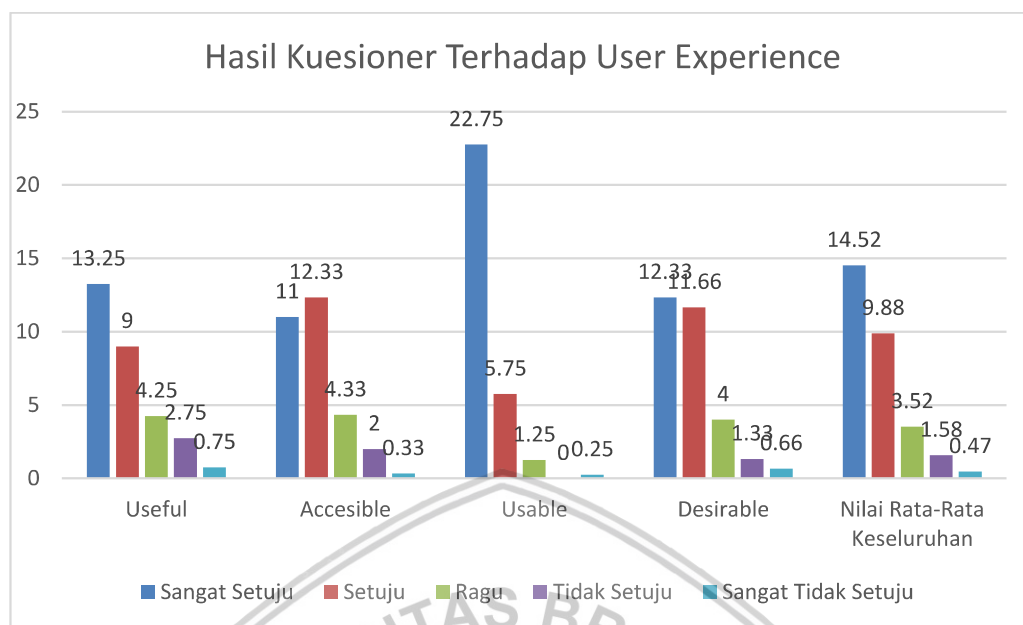
### 6.3.3 Hasil Dan Analisis

Pengumpulan data dilakukan pada lingkungan Universitas Brawijaya, Kos tempat tinggal pembuat kuesioner. Semua responden kuesioner merupakan mahasiswa Universitas Brawijaya Fakultas Ilmu Komputer. Responden diminta secara mencoba langsung alat untuk bermain *game* Minecraft pada *map* yang sudah disediakan dan diminta berjalan melewati rintangan dan menembak target yang telah disediakan seperti pada Gambar 6.14 dimana responden mencoba langsung alat pada Laptop yang telah disediakan.



**Gambar 6.14 Responden Sedang Mencoba Alat**

Dari hasil pengujian alat yang dilakukan terhadap 30 orang Mahasiswa Filkom Universitas Brawijaya dari Jurusan Teknik Komputer dan Informatika menggunakan pertanyaan kuesioner yang terdapat pada lampiran didapatkan hasil sebagai seperti pada Gambar 6.15.



Gambar 6.15 Chart Hasil Kuesioner

Hasil dari Gambar 6.15 tersebut lalu dihitung dengan menggunakan persamaan 6.3 sehingga didapatkan hasil :

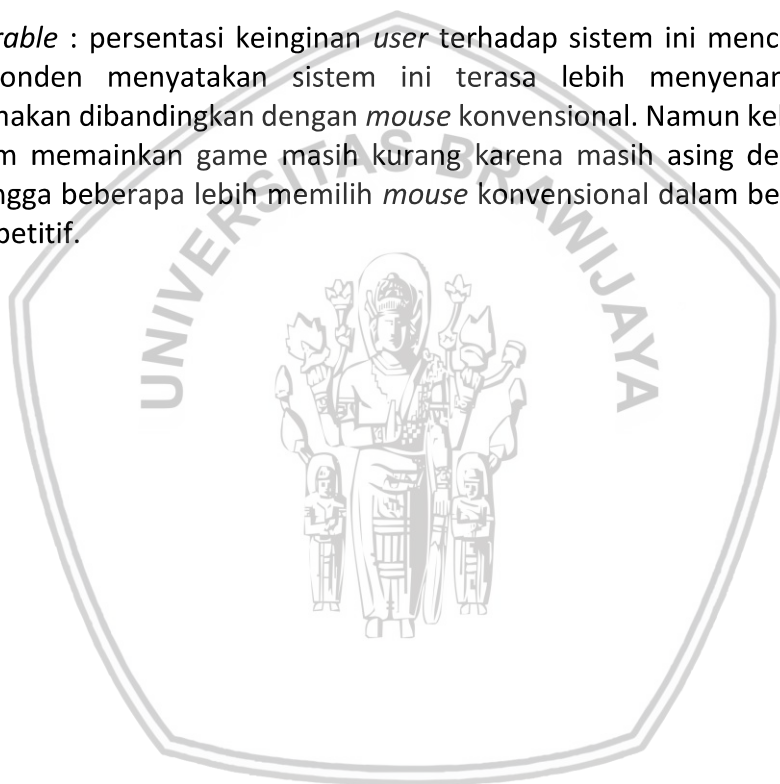
Useful : 76,64%  
 Accesible : 76,38%  
 Usable : 92,29%  
 Desirable : 78,05%  
 Keseluruhan : 80,35%

*Useful* berisi pertanyaan tentang seberapa layak alat ini digunakan dibandingkan dengan *mouse* konvensional untuk mengukur nilai kelayakan dan nilai guna. *Usable* merupakan pertanyaan mengenai fungsionalitas alat terhadap lingkungan *game* dan pengguna. *Accesible* mencakup pertanyaan mengenai seberapa mudah penggunaan alat mulai dari instalasi software komunikasi dengan Mikrokontroller Arduino hingga konfigurasi beberapa bagian fungsionalitas alat. Terakhir, *Desirable* dibuat dengan tujuan mengukur seberapa senang dan keinginan pengguna dalam menggunakan 3D Oriented Mouse Controller dibandingkan dengan *mouse* konvensional.

Dengan nilai keseluruhan dari *user experience* (UX) sebesar 80,35% sistem sudah memberikan nilai kepuasan yang tinggi terhadap kepuasan pengguna. Dari pertanyaan kuesioner yang telah ditanggapi responden didapatkan beberapa kesimpulan terhadap *user experience* yaitu :

1. Useful : persentase seberapa bergunanya sistem ini sebesar 76,64% layak digunakan sebagai pengganti mouse, dengan kekurangan yaitu desain fisik memiliki gagang yang kasar sehingga akan menimbulkan rasa sakit jika digunakan terlalu lama. Namun pada sisi kinerja fungsionalitas sudah layak.

2. *Usable* : persentasi nilai ekspektasi cara kerja sistem yang diharapkan pengguna sesuai dengan produk akhir sistem yang sudah jadi sebesar 92,29%. Permasalah utama dalam desain UX adalah terkadang pengembang produk menganggap pengguna secara umum telah mengerti kondisi teknis dari sistem yang dibuat. Cara kerja sistem dalam menggerakkan *cursor mouse* pada game FPS sesuai dengan yang diharapkan pengguna.
3. *Accesible* : kemudahan konfigurasi dan penggunaan sistem memiliki persentase 76,38%. Hanya dengan menghubungkan sistem dengan ke komputer yang digunakan dan menjalankan *software* sederhana yang berbentuk *executable* tanpa memerlukan instalasi sistem sudah dapat berfungsi normal.
4. *Desirable* : persentasi keinginan *user* terhadap sistem ini mencapai 78,05%. Responden menyatakan sistem ini terasa lebih menyenangkan untuk digunakan dibandingkan dengan *mouse* konvensional. Namun kelincahan *user* dalam memainkan game masih kurang karena masih asing dengan alat ini sehingga beberapa lebih memilih *mouse* konvensional dalam bermain secara kompetitif.



## BAB 7 PENUTUP

Pada bab ini berisi tentang kesimpulan dan saran dari penelitian yang berjudul “Rancang Bangun 3D Oriented Mouse Controller Menggunakan *Finite State Machine* Berbasis Mikrokontroler Arduino”. Kesimpulan mencakup keseluruhan data yang didapat mulai dari perancangan hingga pengujian sistem. Sedangkan saran merupakan ide yang didapatkan oleh penulis ketika telah selesai melakukan pengujian pada sistem.

### 7.1 Kesimpulan

Setelah melewati tahap perancangan, implementasi serta pengujian untuk software dan hardware, dapat diambil kesimpulan sebagai berikut :

1. Membuat *controller* yang memiliki axis 3 dimensi dapat menggunakan sensor *gyroscope* dan *accelerometer*. Data dari sensor ini diubah menjadi satuan *pitch* dan *yaw* sebagai penunjuk arah yang datanya akan diubah menjadi pergerakan pixel dengan matriks x dan y layaknya seperti mouse konvensional. *Pitch* akan mempresentasikan matriks x dan *yaw* mempresentasikan matriks y. *Menggerakkan* mouse pada lingkungan *operating system* dapat dilakukan secara *virtual* dengan mempelajari data apa yang dikirim oleh *mouse* konvensional kepada Windows. Berdasarkan penelitian, *mouse* konvensional hanya mengirimkan data berupa koordinat *output* dari *mouse* tersebut. Perpindahan pixel *cursor mouse* ditangani oleh driver bawaan *mouse* tersebut atau *operating system* sendiri. Oleh karena itu *controller* yang dibuat memerlukan *software* yang berperan untuk menggantikan *driver*.
2. Dalam berkomunikasi dan menerjemahkan data yang dikirim Mikrokontroler Arduino pada PC menggunakan *software* tambahan yang dijalankan pada lingkungan PC. *Software* ini akan membuka *port* serial dan mendengarkan data yang masuk pada *port* tersebut. Arduino Uno memiliki *port type b* sedangkan pada PC jarang terdapat *port* bertipe tersebut. Umumnya PC menggunakan *port* USB, oleh karena itu digunakan kabel USB Type B to USB atau kabel yang umumnya digunakan oleh mesin printer. Dalam berkomunikasi secara serial, *baudrate* pada Arduino Uno dan pada *port* serial PC harus sama. Hal ini bisa ditangani dengan bantuan modul MSCOM dari *operating system* Windows yang direferensikan pada *software*. Dalam komunikasi serial, data yang dikirimkan oleh Arduino Uno dilakukan secara terus menerus. Maka perlu ditambahkan *tag* pada data yang dikirim agar mempermudah *software* dalam mengetahui kapan sebuah sesi data berakhir. Pada sistem ini menggunakan tag “<start>” dan “<end>”. Selanjutnya data tersebut diolah menjadi output yang sesuai melalui *software* dimana *pitch* akan melakukan perpindahan matriks y mouse, *yaw* untuk matriks x, dan 2 button pada sistem untuk melakukan klik kiri dan kanan.



3. Performa sistem yang dirancang memiliki 1,61% error dalam arah yang ditunjukkan *yaw* dan 0,861% error dalam kestabilan nilai saat alat dalam posisi diam. Dalam melakukan pergerakan mouse serta fungsionalitas sistem lainnya terjadi secara *real time* sehingga tidak akan terasa jeda antara pergerakan alat dan pergerakan *cursor mouse*.

## 7.2 Saran

Pada proses penelitian sistem ini hingga selesai, peneliti menemukan beberapa ruang yang masih bisa diolah untuk perkembangan sistem yang lebih lanjut kedepannya berupa:

1. Merancang *hard case* untuk sistem sebaik dibuat bongkar pasang agar mempermudah mengganti komponen yang bermasalah pada sistem. Selain itu *hard case* yang dibuat oleh peneliti menggunakan akrilik ketebalan 0,5cm terasa agak berat, lebih baik menggunakan akrilik dengan ketebalan yang lebih kecil.
2. Pada gagang *hard case* sistem akan menyebabkan rasa sakit pada tangan jika digunakan terlalu lama karena bentuknya persegi panjang. Pada gagang sebaiknya dilapisi bahan yang lembut.
3. Sistem menggunakan sensor MPU-6050 yang memiliki *gyroscope* dan *accelerometer*. *Gyroscope* memiliki *drift* yang memberikan *error* kumulatif dalam jangka waktu panjang. Akan lebih baik jika ditambah modul kompas untuk dikomplementer menggunakan *complementary Filter* untuk menghilangkan *error drift*.
4. Pastikan pada *software* komunikasi yang dibuat memiliki *handle* untuk menutup kembali *port* begitu *software* telah berhenti berjalan. Hal ini akan menimbulkan *error* pada PC ketika *port* yang telah digunakan sebagai jalur komunikasi Arduino Uno dipasang oleh perangkat lain seperti *mouse*.
5. Setiap *engine game* memiliki sensitifitas mouse yang berbeda, oleh karena itu perlu dibuat suatu nilai yang dapat dikonfigurasi oleh pengguna sesuai kenyamanan mereka pada tiap game.
6. Dalam peletakkan kabel jumper pada *hardcase* sebaik dirancang secara rapi untuk mempermudah melakukan pembacaan dan penggantian pada kabel yang bermasalah dan juga mempermudah mengetahui fungsi setiap kabel jumper.

## DAFTAR PUSTAKA

- 3D Transformation*. (n.d.). Retrieved from Tutorialspoint: [https://www.tutorialspoint.com/computer\\_graphics/3d\\_transformation.htm](https://www.tutorialspoint.com/computer_graphics/3d_transformation.htm)
- Baldwin, R. G. (2008, November 18). *Creating a First-Person Shooter Game, Math for Java Game Programmers*. Retrieved from Developer.Com: <https://www.developer.com/java/other/article.php/3785601/Creating-a-First-Person-Shooter-Game-Math-for-Java-Game-Programmers.htm>
- Chua, K. (2017, September 17). *5 most common health concerns for esports athletes*. Retrieved from Rappler: <https://www.rappler.com/technology/features/181571-esports-common-health-concerns>
- Computing the Pixel Coordinates of a 3D Point*. (n.d.). Retrieved from ScratchPixel: <https://www.scratchapixel.com/lessons/3d-basic-rendering/computing-pixel-coordinates-of-3d-point>
- Geometry, Vectors and Transformations*. (n.d.). Retrieved from NRICH Enriching Mathematics: [https://nrich.maths.org/1374#tth\\_sEc2.1](https://nrich.maths.org/1374#tth_sEc2.1)
- Hattenstone, S. (2017, Juni 6). *The rise of eSports; are addiction and corruption the price of its success?* Retrieved from The Guardian: <https://www.theguardian.com/sport/2017/jun/16/top-addiction-young-people-gaming-esports>
- How to convert a 3D point into 2D perspective projection?* (2012, September 17). Retrieved from Stack Overflow: <https://stackoverflow.com/questions/724219/how-to-convert-a-3d-point-into-2d-perspective-projection>
- Hwang, A. D. (2017, Juni 1). *3D projection on a 2D plane ( weak maths ressources )*. Retrieved from Math Stack Exchange: <https://math.stackexchange.com/questions/2305792/3d-projection-on-a-2d-plane-weak-maths-ressources>
- LucasEmanuel. (2013, April 2). *[Tutorial] How to calculate vectors*. Retrieved from BUKKIT: <https://bukkit.org/threads/tutorial-how-to-calculate-vectors.138849/>
- Mouse*. (2017, November 15). Retrieved from Arduino: <https://www.arduino.cc/reference/en/language/functions/usb/mouse/>
- Oosten, J. V. (2011, Juli 6). *Understanding The View Matrix*. Retrieved from 3D Game Engine Programming: <https://www.3dgep.com/understanding-the-view-matrix/>

- Paradise, A. (2017, Februari 17). *The importance of streaming to e-sports*. Retrieved from Tech Crunch: <https://techcrunch.com/2017/02/17/the-importance-of-streaming-to-e-sports/>
- Paradise, A. (2017, februari 17). *The importance of streaming to e-sports*. Retrieved from TechCrunch: <https://techcrunch.com/2017/02/17/the-importance-of-streaming-to-e-sports/>
- Richmond, M. (2016, September 15). *When you fire a scoped rifle, why does the bullet always hit where the scope crosshairs are placed if the scope is above the barrel?* Retrieved from QUORA: <https://www.quora.com/When-you-fire-a-scoped-rifle-why-does-the-bullet-always-hit-where-the-scope-crosshairs-are-placed-if-the-scope-is-above-the-barrel>
- Robert Mahony, T. H.-M. (2006). A Study of non-linear complementary filter design for kinematic systems on the special orthogonal group. *Internal Report : ISRN I3S/RR-2006-36-FR*, 1-4.
- Rodenas, L. (2014, Januari 7). *Arduino Sketch to automatically calculate MPU6050 offsets*. Retrieved from I2Cdevlib Forums: <https://www.i2cdevlib.com/forums/topic/96-arduino-sketch-to-automatically-calculate-mpu6050-offsets/>
- Utkarsh Tiwari, V. S. (2016, Oktober 11). *Gesture Mouse*. Retrieved from Arduino Project Hub: <https://create.arduino.cc/projecthub/onyx/gesture-mouse-a4320a>
- Vance VanDoren, P. P. (2014, 8 28). *Open- vs. closed-loop control*. Retrieved from Control Engineering: <https://www.controleng.com/single-article/open-vs-closed-loop-control/d62442ec45d64b0e41b012e849b397a8.html>

## LAMPIRAN A : HASIL KUESIONER

No	Pertanyaan	Sangat Setuju	Setuju	Ragu	Tidak Setuju	Sangat Tidak Setuju	Nilai
1	Apakah penggunaan 3D Oriented Mouse dalam bermain game berbasis FPS sudah sesuai ? (Kesesuaian)	24	3	3	0	0	92.5
2	Apakah penggunaan 3D Oriented Mouse sudah layak dibandingkan dengan <i>mouse</i> konvensional ? (Kesesuaian)	8	20	2	0	0	80
3	Apakah penggunaan 3D Oriented Mouse sudah memberikan output yang sama seperti <i>mouse</i> Konvensional ? (Kesesuaian)	17	8	5	0	0	85
4	Dalam bermain game FPS dalam tingkat kompetitif, apakah anda lebih memilih menggunakan 3D Oriented	4	5	7	11	3	46.66

	Mouse Controller ? (Kesesuaian)						
5	Apakah kestabilan dari 3D Oriented Mouse Controller sudah bagus ? (Teknis)	5	14	7	4	0	66.66
6	Apakah Software untuk melakukan konfigurasi pada 3D Oriented Mouse sudah mencakup semua kebutuhan yang diperlukan alat ? (Teknis)	9	17	3	1	0	78.33
7	Apakah penggunaan <i>Joystick</i> pada 3D Oriented Mouse Controller memberikan kendali sesuai dengan yang diharapkan ? (Teknis)	22	8	0	0	0	93.33
8	Apakah dalam menggerakkan cursor menggunakan 3D Oriented Mouse	7	17	4	0	2	72.5



	Controller sesuai dengan yang diharapkan ? (Teknis)						
9	Apakah dalam melakukan perputaran pada game menggunakan 3D Oriented Mouse Controller sesuai ? (Teknis)	5	7	11	7	0	58.33
10	Dalam bermain game FPS, apakah menurut anda 3D Mouse Controller lebih akurat dalam penggunaannya ? (teknis)	18	11	1	0	0	89.16
11	Apakah dalam melakukan konfigurasi 3D Oriented Mouse ( melalui software dan hardware) sudah mudah ? (Kemudahan)	19	11	0	0	0	90.83
12	Pada saat bermain game berbasis FPS, apakah dalam mengendalikan pergerakan	30	0	0	0	0	100

	dalam game menggunakan <i>Joystick</i> sudah mudah ? (Kemudahan)						
13	Pada saat bermain game berbasis FPS, apakah dalam mengendalikan cursor mouse menggunakan 3D Oriented Mouse Controller sudah mudah ? (Kemudahan)	15	9	5	0	1	80.83
14	Dalam bermain game berbasis FPS, apakah dalam penggunaannya dari segi pergerakan karakter dan mouse, 3D Oriented Mouse Controller lebih mudah dibandingkan menggunakan Keyboard dan Mouse Konvensional ? (Kemudahan)	27	3	0	0	0	97.5
15	Apakah lebih menyenangkan jika bermain game FPS menggunakan 3D Oriented Mouse	21	9	0	0	0	92.5

	Controller dibandingkan menggunakan <i>mouse</i> konvensional ? (perasaan)						
16	Apakah lebih nyaman jika bermain game FPS menggunakan 3D Oriented Mouse Controller dibandingkan menggunakan <i>mouse</i> konvensional ? (perasaan)	4	17	8	1	0	70
17	Apakah lebih mudah jika bermain game FPS menggunakan 3D Oriented Mouse Controller dibandingkan menggunakan <i>mouse</i> konvensional ? (perasaan)	12	9	4	3	2	71.66
	Rata-rata	14.52	9.88	3.52	1.58	0.47	80.34